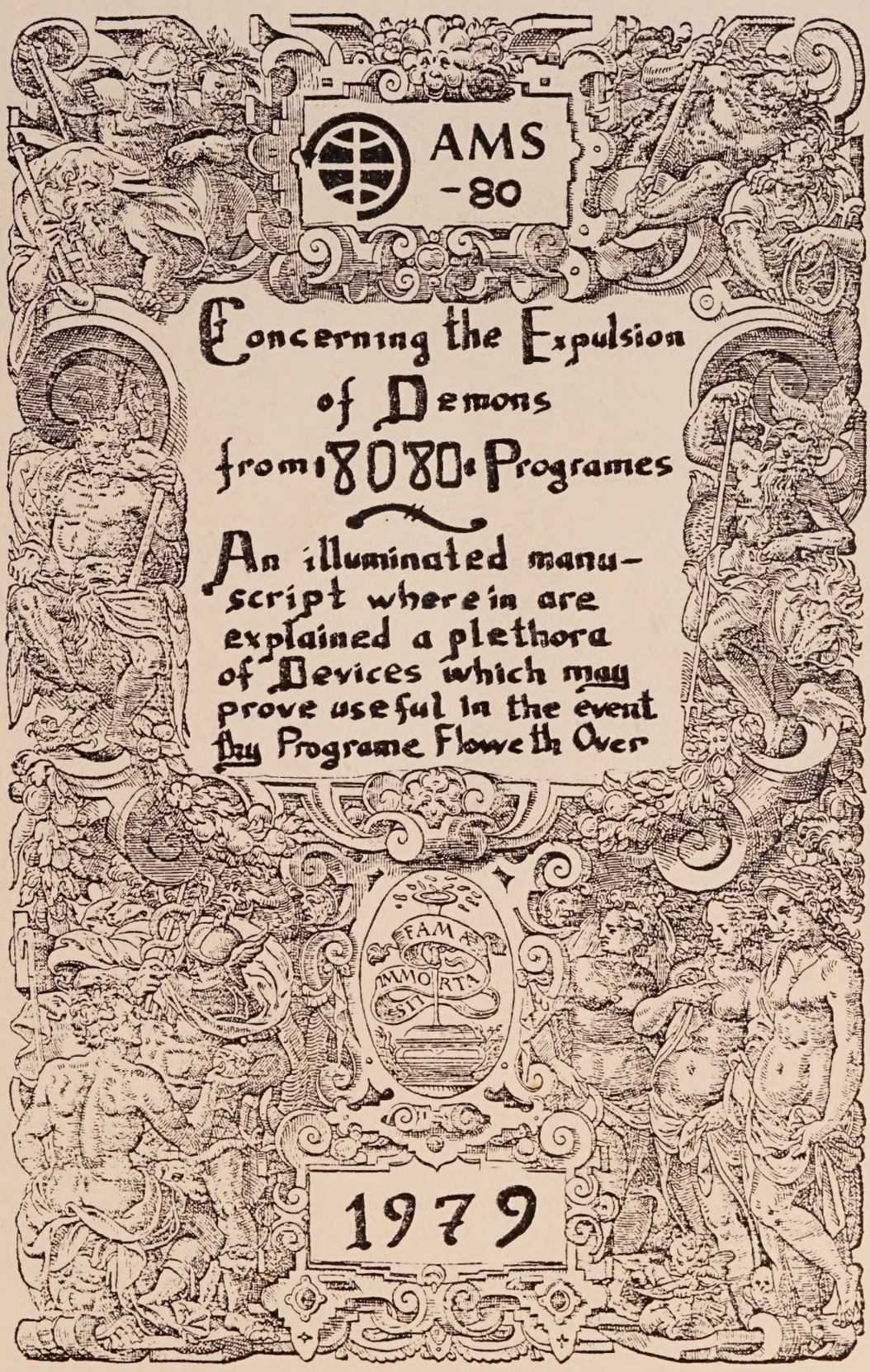

AMS
-80
Concerning the Expulsion
of Demons
from 8080 Programes
An illuminated manu-
script wherein are
explained a plethora
of Devices which may
prove useful in the event
thy Programe Flowe th Over
FAM Æ
IMMO RTA
SIT
1979

REV 5.8

Digitized by the Internet Archive
in 2025 with funding from
Amateur Radio Digital Communications, Grant 151

https://archive.org/details/ams-80doc

AMS-80 DOCUMENTATION
Version 5.8.
© 1979 Snow Micro Systems Inc.
P.O.B. 1704
Silver Spring,
Md. 20902. USA.

S/N H MAGNUSKI
For 103

TABLE OF CONTENTS

1. Introduction 1

2. AMS80 Command Descriptions 3
   Command Summary List 11

3. Software Routines and Utilities 12

4. The Custom Section of AMS80 -- Description 18
   Customizing AMS80 20

5. VBD4 Video Driver Software -- General Description 21

APPENDICES

A. AMS80 Listing A-1
   Interface Jump Table A-3
   Assignable I/O Jumps A-6
   "Hard-Core" Code A-44
   Hex Dump

B. System Info:
   Suggested/Typical Memory Map With AMS80 B-1
   Typical System I/O Allocation B-2
   Ram Reserved Words Required byAMS80 B-3
   CP/M Compatibility B-4

C. Sample Customization Examples C-1

H. Hints & Kinks: A Veritable Curiosity Shoppe of
   Misc. & Assorted Trivia Vaguely Applicable to AMS80 H-1
   Phase II Satellite Orbital Program H-12

N*. Sample Interfaces Between AMS80 and NORTH-STAR N*1

V. VDB4 Video Driver:
   Operating and Interface Documentation V-1
   TTY-Like Text-Mode Driver V-4
   Graphics Driver V-9
   BASIC-Callable Graphics Interfaces V-11
   VDB-4 Hex Dump V-11

X. Hardware Data
   Suggested Modifications and Strapping For SSM IO-4 Board X-1
   Suggested Modification For Fully Programmable Serial-B Port
   For SSM IO-4 Board X-2

1. INTRODUCTION

AMS-80 is a PROM-based monitor intended to be permanently resident in your 8080 system. AMS-80 is not intended to be a be-all and end-all operating system, but rather it is intended to fulfil the following major functions:

A. Allow the user to exercise control over all the major functions in the system in a manual mode. The user can examine locations in memory in hex, ASCII or symbolic (assembler-type) mnemonics. The user can change any byte of memory, can copy memory from one location to another, fill memory with constants or ASCII text. The user can verify that RAM memory is OK with a simple, non-destructive test. The user can force data to be inputted or outputted to any I/O port.

B. The user may execute any program in memory, and if that program is in RAM, he may set two software break-points. After a breakpoint is encountered, the user may examine any of the internal 8080 registers and change them if desired, and then continue execution.

C. AMS-80 makes available to the user a well-defined, controllable I/O protocol for printer/reader/punch/console devices and allows him to dynamically re-assign these devices if desired. These features are independent of the specific device employed; for example, printer-type output could be directed to a teletype, a CRT or a line printer, while input could come from a console terminal, a high-speed reader, etc.

D. Included in the structure of AMS-80 is a powerful 64x16 video display driver which supports several modes of operation when displaying text; the video display can be configured to list a page at a time, or to continually scroll, while some lines of text at the top of the screen can be protected from normal erasure. Limited graphics are also supported.

E. AMS-80 helps users to make their software transportable. All I/O or utility entries from the user's programs can be directed through a jump table. To transfer software from user to user, all that is required is a standardized jump table; the user is free to put his code anywhere. If later revisions of AMS-80 relocate software in the monitor ROM, the user will be buffered from these revisions since the jump table will remain the same.

F. AMS-80 assumes a hardware configuration compatible with many, but not all hobby systems. The memory map given on page B-1 is compatible with Processor Technology, IMSAI, North Star and many other common systems. AMS-80 I/O is directly

compatible to ZAPPLE. Thus, for example, CP/M configured for ZAPPLE will run through AMS-80 with no changes. 8080 systems known to be incompatible include the Heath H-8, Polymorphics, Intel SBC and, of course, all non-8080 machines. Z-80 systems should run OK, except that the symbolic disassembler will only work on the 8080 sub-set of instructions, and the register examine/modify feature only works on one of the two sets of internal registers.

CAVEATS:

Since the authors have developed this software for the hobbyists, and not as a commercial product, there may be a few bugs. Please let us know and we will do our best to correct them. This documentation is not intended as a textbook on 8080's so you may find some things less than perfectly clear; again, let us know. Rather than delay this documentation, we have liberally annotated various portions by hand, under the thought that the hobbyist would prefer to have the information in a less-than picture perfect copy to having an incomplete document. We also appologyze for speling errurs.

AMS-80 and AMSAT:

Both the authors are active in AMSAT, an acronym for the Radio Amateur Satellite Corporation. This world-wide group has built and launched four of the OSCAR (Orbiting Satellite Carrying Amateur Radio) satellites in the past 10+ years. This group is perhaps the most advanced user of hobby computers in the amateur radio discipline. Even in the early 70's, our members were using basement machines to command and control our satellites, to make orbital computations, etc. We have structured AMS-80 to provide a cohesive tie between the individuals who are working on this program. For you non-ham users of AMS-80, this may account for such features as the Baudot-ASCII conversion routines, as well as the name

AMS-80.

If you wish further information on AMSAT, even if you are not a licensed amateur radio operator, write

AMSAT
P. O. Box 27
Washington, D.C. 20044

P. S. This copy of AMS-80 documentation includes revisions through August, 1979, and covers revision 5.8 of AMS-80 and the VDB-4 video driver.

T. Clark - W3IWI

The AMSAT motto:

WE, THE WILLING, ARE DOING THE IMPOSSIBLE
FOR THE UNGRATEFUL !!
WE HAVE DONE SO MUCH FOR SO LONG WITH SO LITTLE,
THAT WE ARE NOW QUALIFIED TO DO ANYTHING
WITH ABSOLUTELY NOTHING ! ! ! !

2. AMS-80 COMMAND DESCRIPTIONS

A Command -- ASCII MEMORY DUMP

Format A Low Address, High Address

Upon execution of this command, data bytes located in the block of memory bounded by the low and high addresses are displayed at the console as ASCII characters. Up to sixteen characters per line are displayed preceeded by the hexadecimal address of the first byte of that line.

B Command -- BASIC

Format B

The B command is used to execute an immediate jump to an already resident BASIC, etc. For North-Star, BASIC resides at 2AØØH (single density) or 2DØØ (double density), and you should be certain to code the customization area (at FC59H) with the appropriate jump address. Feel free to use "B" for any other custom requirement you may have.

C Command -- CONFIGURE I/O DEVICES

Format C Device

This command is used to configure the I/O devices to particular physical units. The devices are (P, L, R, C and V). Upon execution of the command the current configuration is displayed at the console. The configuration may be changed by entering a new digit between 0 and 3. If no change is desired, enter a carriage return character.

D Command -- DISPLAY MEMORY DATA

Format D Low Address, High Address

Upon execution of this command data located in memory between the low address and the high address are displayed at the console. Up to 16 bytes per line are displayed, preceded by the hexadecimal address of the first byte of that line.

E Command -- END FILE ON TAPE

Format E Address

The E Command causes an end of file mark and sixty null characters to be punched at the end of a hexadecimal output file. The end of file mark is a hexadecimal record of length ØØ. If the address is Ø or absent, the R Command that loads the tape will return control to AMS-80. If the address is non-zero, the R Command will transfer control to that memory location immediately after loading the file.

F Command -- FILL MEMORY WITH CONSTANT

Format F Low Address, High Address, Constant

Execution of this command causes the block of memory bounded by the low address and the high address to be filled with the constant.

G Command -- GO TO MEMORY LOCATION

Format G Address, bkpt 1, bkpt 2

The G Command causes program control to be transferred to the address. This is useful for starting programs. If either bkpt 1 or bkpt 2 is specified, a breadkpoint will be set in the program at the corresponding address. The specified address must correspond to the first byte of an instruction. If either breadpoint is encountered during program execution, AMS-80 will save all program status (CPU registers and condition bits) clear all existing breakpoints and take control. The user may then examine and/or modify registers or memory, or use any other command. This feature allows the user to exercise and debug both hardware and software.

If an address is not specified, the program status is restored and the saved value of the program counter is used as the new starting address. The "@" command works just like "G" except that it re-enables AMS-80's default interrupt state (optional command)

H Command -- HEXADECIMAL ARITHMETIC

Format H Number #A, Number #B

This feature performs hexadecimal arithmetic on the numbers A and B. Upon termination of the entry sequence, the results are printed out as follows:

(A + B) (A - B) (B - A)

This feature is useful for computing offsets or relative locations.

I Command -- INPUT DATA (I/O PORT),

Format I Port Number,

Upon execution of the command, AMS-80 does an input from the specified I/O port and displays the input data at the console as a hexadecimal number.

J Command -- JAM TEXT

Format J Address

The J Command is used to JAM ASCII text into memory, starting at the location specified. A carriage return character terminates the sequence. At termination, an end-of-text byte (FF) is written into the memory byte immediately following the text string, thus allowing the string to be printed out using the MSG sub-routine. If an error is made during entry, the Control-H character may be used as a backspace character to overwrite the erroneous entry.

K Command -- Not implemented. Free for your use.

L Command -- LANGUAGE DISPLAY (SYMBOLIC DIS-ASSEMBLER LISTING)

Format L Address

The L Command is used to display the contents of a program in memory in symbolic assembly language at the console. The readout is stopped by depressing any key on the console keyboard. Note that data tables, un-implemented instructions, etc. produce "garbage" display characters. All 8080 instructions are supported, but the 8085 RIM/SIM and all Z-80 "special" instructions are not.

M Command -- MOVE MEMORY

Format M Low Address, High Address, New Address

The M Command is used to move the block of data residing in memory between the low address and the high address to a new area of memory starting at the new address. The data is unchanged, so that the new area of memory contains a copy of the data in the initial area. If the new and initial areas overlap, the results may be undesired.

N Command -- PUNCH NULLS ON TAPE

The N Command caused 60 null characters to be punched. This is six inches of blank paper tape.

O Command -- OUTPUT DATA (I/O PORT)

Format O Port #, Data

The O Command causes AMS-80 to output the data to the specified I/O port.

P Command -- PARSE TAPE

Format P

Execution of the P Command causes data to be read from the Reader device and output to the Punch device on a character by character basis. This allows tapes to be copied directly. If, however, the Console is assigned to the Punch, the contents of the tape will be displayed at the console. This allows tapes to be scanned for particular programs or data.

Q Command -- OPERATION SYSTEM START ("QUIT")

Format Q

The Q Command is reserved for a custom patch so as to execute a jump to an operating system. As distributed, Q disables interrupts and performs an immediate JUMP to North-Star's E800 BOOTSTRAP address.

R Command -- READ HEX FILE FROM TAPE

Format R Bias Address

This command loads a paper tape punched in INTEL hexadecimal format (using the W Command) into memory. The address at which the tape is loaded is determined by adding the address punched on the tape to the bias address using two's complement arithmetic. The bias may be negative, but must be in two's complement form. If the tape was produced using an E Command with a non-zero entry point address, control will be transferred to that location in memory. Otherwise AMS-80 will remain in control and request another command. If no bias address is entered, zero will be assumed and the data will go into memory at the location specified by the tape.

If an error occurs while reading the tape, AMS-80 will immediately stop reading the tape and indicate that it is awaiting the next command. The operation may be retried by backing up the tape to any point before the last colon and issuing another R Command.

Note that this means that if a program is changed, it is not necessary to regenerate an entire new tape with the change: instead, the original tape may be read in followed by a patch tape that changes the contents of "erroneous" memory locations.

S Command -- SUBSTITUTE MEMORY

Format S Address

The S Command is used to display and/or modify the contents of individual memory locations. It is useful in manually entering a patch into a program. It is used as follows:

a. Type "S address comma".

b. The data from the selected memory location is displayed.

c. To modify the contents of the memory location, type in new data followed by a "comma". If you do not wish to modify the data, type a "space" or a "comma".

d. If a "space" or a "comma" character was typed in step c, the contents of the next memory location will be displayed and may be operated upon as in step c. If a "carriage return" character was typed, control will be returned to AMS-80.

T Command -- TITLE TAPE

Format T

Upon entry of the T, AMS-80 will echo an "=" sign. ASCII text may then be entered at the console. The sequence is terminated by a carriage return character. This function is used to put titles or headers on hexadecimal tapes.

U Command -- I/O CONFIGURATION "UNKNOWN"

Format U

The U command causes AMS-80 to display the current I/O configuration at the console as follows:

C = n R = n P = n L = n V = n

where n is the physical device configured to the particular category of I/O device. Changes may be made by the use of the C command. The number associated with the V, indicates the mode assigned to the Video Display board (see sections 5 and V).

V Command -- RAM MEMORY VERIFICATION

Format V Low Address, High Address

The V Command tests a block of RAM memory bounded by the low and high addresses. Any "bad" locations are listed. This Command may be used to checkout RAM Circuit Cards. This is only a cursory test and doesn't do a good job at finding "wired-or" solder bridges or "dynamic" problems!

W Command -- WRITE THE CONTENTS MEMORY TO TAPE

Format W Low Address, High Address

The W Command is used to output the contents of consecutive memory locations starting from the low address and continuing to the high address to the punch device in hexadecimal format. A series of W Commands may be used in order to punch various non-contiguous memory locations onto a continuous strip of tape.

Any series of W Commands should be terminated with an E Command in order to punch an end of file mark so that the R Command will function correctly.

X Command -- EXAMINE/MODIFY REGISTERS

Format X Identifier

The X Command is used to display and/or modify the CPU registers. It is the only way of accessing the contents of the internal registers of the CPU. It operates in a similar manner to the S Command, as follows:

a) Type an X followed by the Register identifier listed below:

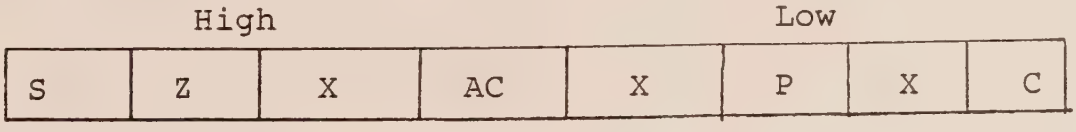
A = A register
B = B register
C = C register
D = D register
E = E register
H = H register
L = L register
S = Stack Pointer
P = Program Counter
M = H and L registers combined

F = Flag byte, displayed in the form as it is stored by the PUSH PSW instruction

No identifier, displays all the registers in sequence.

b) The contents of the selected register are displayed followed by a dash (-). Four hexadecimal digits are displayed for the M, P and S Registers; two hex digits for the others.

The format of the Flag byte F is as follows:



S = Sign Bit
P = Parity Bit
Z = Zero Bit
C = Carry Bit
AC = Aux Carry Bit
X = This bit may depend on the Manufacturer of the 8080.

c) To modify the contents of the selected register, type in the new data followed by a "comma" or "carriage return" character. If you do not wish to modify the register, type in only the "comma" or "carriage return" characters.

d) If a "space" or a "comma" character was typed in step c, the contents of the next register in alphabetical order is displayed. If a "carriage return" or a "comma" character (after displaying the contents of the S register only) is typed, the command is terminated and control is returned to AMS-80.

Z Command -- ZIP TO DOS

The North-Star - AMS-80 software distributed by W3IWI (see Appendix N*) has 29ØØH as an initialization entry point. As distributed, Z does a jump to 29ØØH.

Notes

(1) Execution of the A, D, L, P, and V Commands may be interrupted by depressing any keyboard character at the console. The display will terminate at the end of a line.

(2) With the sole exception of the S command, when advancing to the next memory location without changing the contents of the current location, all entries ignore "space" characters completely. The delimiter character, thus, should be the "comma" and not the "space".

TABLE 1. AMS-80 Version 5.8 Command List Summary

A	PRINT (MEMORY) IN ASCII
B	JUMP TO LOADED BASIC OR OTHER USER SOFTWARE
C	CONFIGURE I/O DEVICE
D	DISPLAY (MEMORY) IN HEX
E	WRITE END OF FILE RECORD TO TAPE
F	FILL (MEMORY) WITH CONSTANT
G	GO TO LOCATION AND EXECUTE (WITH BREAKPOINTS AVAILABLE)
H	HEX MATH (SUM & DIFFERENCES)
I	INPUT FROM PORT TO CONSOLE
J	JAM TEXT INTO MEMORY
K	*
L	PRINT DISASSEMBLED (MEMORY) IN ASSEMBLY LANGUAGE LISTING FORMAT
M	MOVE BLOCK OF (MEMORY)
N	PUNCH 6 INS LEADER TAPE NULLS
O	OUTPUT TO PORT FROM CONSOLE
P	PARSE AND COPY TAPE
Q	QUIT AND JUMP TO DISK OR OTHER OPERATING SYSTEM
R	READ TAPE
S	EXAMINE/SUBSTITUTE (MEMORY)
T	PUT HEADER ON TAPE
U	DISPLAY I/O CONFIGURATION "UNKNOWN"
V	VERIFY RAM MEMORY BLOCK
W	WRITE TO TAPE
X	EXAMINE/CHANGE (REGISTERS)
Y	*
Z	ZIP TO ALREADY LOADED NORTH-STAR DOS

* Not assigned as yet.

Details of the operation of the commands are given in the description of AMS-80.

3. SOFTWARE ROUTINES AND UTILITIES

The jump table interfaces AMS-80 and its utilities to the User. This allows later versions (if any) to be incorporated in a system, without requiring any changes in user programs, because although the internal locations of routines may change, the jump table location does not. Since there is no "Jump Indirect" instruction in the repertoire of the 8080, the Jump table format has been chosen to implement the function. Each routine will now be described in turn.

	AMS-80 ENTRY POINT ADDRESS
CUSTOM This routine sets up all the programmable I/O devices (eg. 8251 and 8255's) in the system as well as any flags or buffers (eg. cassette I/O). It is used during an initial power up phase.	= FØØØ
CONSI. This routine inputs a character from the system Console to the accumulator of the 8080. Note that all eight bits are input.	= FØØ3
RDR This routine inputs a character from the reader device to the accumulator as a single byte.	= FØØ6
CONSO This routine outputs the contents of the C Register to the System Console device as a single byte.	= FØØ9
PUNCH This routine outputs the contents of the C Register to the Punch Device as a single byte.	= FØØC
LIST This routine outputs the contents of the C Register to the List Device as a single character.	= FØØF
CSTS This routine tests the console to see if a key has been depressed. It returns with the A Register set as follows: A = FF if key has been depressed. A = 00 if key has not been depressed.	= FØ12

IOCHK

Returns the contents of the I/O configuration control word (word 3) in the accumulator. = FØ15

IOSET

Deposits the contents of the C Register into the I/O configuration control word (word 3). = FØ18

MEMCK

Returns the address of the top of user RAM area in the following configuration: = FØ1B

B = high order byte.

A = low order byte.

RESTART

The breakpoint entry. The contents of all the CPU registers are saved, and control is returned to AMS-80. The address at which program execution was interrupted is printed at the console. = FØ1E

START

The entry point to the AMS-80 main command loop. Programs can be written such that they return to AMS-80 without resetting any flags or I/O devices. = FØ21

BEGIN

Restart AMS-80, by resetting the stack but bypass the custom area so that the system configuration does not change. = FØ24

CHIN

This routine waits for a key to be depressed at the console, echos it to the display, and returns with the character in the accumulator. = FØ27

AOUT

This routine outputs the contents of the accumulator to the system console as a single character. = FØ2A

TCSTS

This routine tests the system console keyboard. If a character has been depressed, it returns control to AMS-80; if not, it returns to the calling program. This routine is very useful when programming small routines by hand. It allows the keyboard to interrupt the routine and return control to AMS-80. = FØ2D

TCRET

A carriage return, line feed sequence is output to the console. = FØ3Ø

THXN

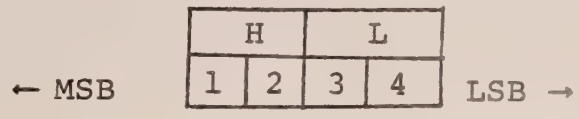
The contents of the least four bits of the accumulator are converted to ASCII and output to the console device. This routine may be used to print BCD or similar data at the console. = FØ33

THXB

The contents of the accumulator are printed out on the console device as two hexadecimal digits. = FØ36

THXW

The contents of the H and L registers are output to the console as four hexadecimal digits. The sequence is as follows: = FØ39



MSG

This routine prints an ASCII message stored in memory. The contents of the H and L registers must point to the first byte of the text. The message must be terminated by an end of text byte (OFFH). For an example of the use of the routine, see the sign on message text at location F139 (Hello) called by the routine at F196. = FØ3C

PCHK

This routine tests for delimiters, spaces and commas are allowed. The carry flag is set by any non-valid delimiter character. = FØ3F

CONSB

The contents of the B register are output to the console as a single byte. = FØ42

PHXB

The contents of the accumulator are output to the punch as a single byte. = FØ45

LEAD

Punch 6 inches of leader tape on the punch device. = FØ48

PCRET	Output a carriage return-line feed sequence to the system punch device.	= FØ4B
PHXW	Output the contents of the H and L register to the system punch using the same format as THXW.	= FØ4E
POB	The contents of the B register are output to the punch as a single byte.	= FØ51
LHXW	as THXW but to the list device.	= FØ54
LHXB	as THXB but to the list device.	= FØ57
LCRET	as TCRET but to the list device.	= FØ5A
LOB	as POB but to the list device.	= FØ5D
RHXB	Read two consecutive hexadecimal characters from the reader device and return with them packed as two four bit hexcharacters. When used in the tape reading routine, the check sum is updated.	= FØ6Ø
RIX	A character is read in from the reader device to the accumulator. The most significant bit (bit 7) is masked to zero.	= FØ63
CONV	Converts the contents of the accumulator from HEX (or BCD) to ASCII. Returns with the ASCII character in the C register.	= FØ69
NIBBLE	Converts the ASCII character in the accumulator to a HEX value. Returns with the carry flag set, if the character was not a valid HEX value.	= FØ6C

*

*Note:
FØ66 has room for a new/special command jump.

DONE

Compares the contents of the H and L register pair with the contents of the D and E register pair. The H and L register pair is incremented during this routine. The carry flag is set if the contents of the H and L register pair are zero, or if the contents of the D and E register pair have a value greater than that of the contents of the H and L register pair. = FØ6F

TIMER

This is a delay routine. The contents of the D and E register pair are decremented to zero before returning to the calling routine. This routine is used in software timing loops. Depending on your clock speed and wait states in AMS-80's EPROM, a <DE> value of FFFFH will give a delay of ½-3 seconds. You should "calibrate" your system if you use this utility. = FØ72

SDEHL

The value of the contents of the D and E register pair is subtracted from the value of the contents of the H and L register pair. The result is returned in the H and L register pair. = FØ75

LOCMB

This routine is an indirect locate routine. The contents of the D and E registers are added to the base value of the block of user defined RAM associated with AMS-80. The result is returned in the H and L registers so that memory register instructions can be used. This technique is chosen for locating flag bytes to minimize the different customizing requirements for different systems. The first four bytes of this table are reserved for use by AMS-80 Rev. 5.8. = FØ78

IRST

This routine enables the 8080 interrupts and resets the 8214 programmable interrupt integrated circuit. As supplied, the output port assigned to the Interrupt Control Port is whatever happened to be in the system master at the time the ROM was burned and the listing was made!* The contents of the A register = FØ7B

*But may be changed at byte FB45 if your system is different For example, the WAMECO CPU board uses Port = DF (Despite their documentation that says BF).

must be set to the compliment of the level of interrupt being re-enabled. If no priority requirements exist, set it to Ø2 so that RSTØ and RST1 interrupts are "hot".*

BACON

The contents of the accumulator are converted from Baudot to ASCII. = FØ7E

ASCBD

The contents of the C register are converted from ASCII to Baudot. If an illegal Baudot character was input to ASCBD, the carry flag is set upon return. Bit 7 is set if a LETTER shift is required, and Bit 6 is set if a FIGURES shift is required. A sample routine using ASCBD is presented in the I/O Hints & Kinks section (see BAUDO). = FØ81

**

*The initialization default = Ø2, but may be changed to meet your requirements at F18FH and FBFØH.

**Note:
Locations FØ84 & FØ87 may be used for customized jumps.

4. THE CUSTOM SECTION

The custom section begins at location FC00, with the jump tables to the I/O device drivers for each of the four categories of devices (Punch, reader, list and console). The routines operate in the following manner:

PUN	outputs the contents of the "C" register to the selected punch device.	= FCØØ-FCØB
RDR	inputs one character (8 bits) from the selected reader device to the "A" register.	= FCØC-FC17
LIST	outputs the contents of the "C" register to the selected list device.	= FC18-FC23
CONO	outputs the contents of the "C" register to the selected console device.	= FC24-FC2F
CONI	inputs one character from the selected console device to the "A" register.	= FC3Ø-FC3B
CSTS	tests the state of the selected console device keyboard to determine if a key has been depressed. CSTS returns with 00 in the "A" register if no key has been depressed, and FF in the "A" register if a key has been depressed.	= FC3C-FC47

In all of the above routines the actual physical device is determined by the I/O configuration of word 3 (IOCW). All of the entry or jump locations are located between memory addresses FC00 and FC47.

The number shown in the sample listing associated with each device (e.g. PUN3, RDR2, RDR0, CONI3, etc.) is the device number recognized by the AMS-80 C command and displayed by the AMS-80 U command.

BTTI	is the entry point to a routine that drives a Baudot terminal keyboard or RTTY terminal unit and returns an ASCII equivalent character in the "A" register.	= FC48

BTTO is the entry point to a routine that converts the ASCII character in the "C" register to its baudot equivalent and outputs it to a serial baudot terminal device such as a Model 15 Teletype™ or an afsk unit. = FC4B

TBSE must contain the base address of a block or RAM that can be used by the standard section of AMS-80 as well as the custom area for buffers and flag locations. This block of RAM should preferably be well separated from the main user memory area. = FC4E

TPMR contains the value (X + 100H) where X is the address in hex of the top of the RAM area. The stack is automatically set up here. = FC51

IORW is a single byte containing the default state of the I/O configuration. The I/O configuration is set to this value on power up of the system. = FC5Ø

TPMR1 contains the value (Y + 100H) where Y is the address in hex of the top of user area of RAM. = FC53

Note that TPMR and TPMR1 do not have to be set at their actual values in any system. They should rather be used to guard non available areas. For example, in a system having a video display board with memory mapped I/O, TPMR1 should be set to the value shown in the sample listing, thus ensuring that the video memory area is bypassed when user programs use AMS-80 to find the top of available memory. If a block of RAM is not present above, the video area TPMR should be set to the same value as TPMR1 else AMS-80 will set the stack up in the video RAM area, the sign on message will then clear the video RAM area and blow the system. This buffer RAM is, as explained elsewhere, reserved for buffers and flags. If it is well separated from the main RAM area, the buffers can be used with commercial software such as BASIC without any or too many patches to that software.

VECTOR contains the command table. A total of 27 commands (@-Z) are available out of which two are unassigned. This table resides from FC55H to FC8AH.

When adding custom commands, remember that AMS-80 is a debug package, not an operating system. In the listing it will be seen that the standard commands branch to absolute locations within the standard AMS-80 block while the custom ones either branch to an error location (not assigned as yet) or to an address (symbolic) within the custom area.

CUSTOM at FC8B contains the custom device initialization routines. This area sets up programmable I/O devices such as 8255's and 8251's, and any buffer pointers that have to be initialized at turn on.

IT MUST END WITH A JUMP TO LOCATION BEGIN = F14D, AND IT MUST ALSO CONTAIN THE CODE TO TRANSFER THE CONTENTS OF TBSE TO MEMORY LOCATION 000EH.

This section is only used when AMS-80 is first booted up or hen it is reset by entry at FØØØ. Note that AMS-80 initializes locations Ø-2 with a jump FØØØ (C3 ØØ FØ), so any time you hit RESET, this code is executed!

USER contains the jumps to the device I/O drivers in the system.

NOTE that the order of device assignments and memory addresses between PUN3 and the USER custom initialization MUST NOT BE CHANGED (e.g. FCØØ-FC8B).

Customizing AMS-80

In order to customize AMS-80 it is advisable that a system memory map be generated and the assignments for TBSE, TPMR and TPMR1 be made accordingly. A sample map is shown on page 21 IORW must also be set up at this time. The pointers to the RAM blocks should be set up as previously described. An I/O assignment table should also be generated as shown on page B-2 and the value of IORW determined. If four devices are not present in any system, there is no reason why a physical device cannot be assigned to more than one logical device or that a physical device should not be assigned a RAM location in the RAM block between 40 and 100 or up in the RAM defined by TBSE. This will allow routines to be developed for new devices and tried out before burning in EPROM. It will also allow them to be read into the system from tape if an EPROM zapper is not readily available.

The video control byte at ØØØBH must be set if the VDB4 video display module software is used. The custom area must end with the following code:

```
LHLD  TBSE
SHLD  ØØØEH
MVI   M,Ø
...   ...
JMP   BEGIN
```

These lines of code set up the pointer to RBSE, clear the case memory for the ASCII/Baudot and Baudot/ASCII conversions and then return the execution to the standard section.

Sample Customization:

Appendix C gives a sample of a typical customization. Unless you "negotiated" a special customization, this is the code on your ROM. A second full customization (W3IWI's current system) is included to give you a few ideas. Other customization features and I/O utilities which have been found useful are included in Appendix H.

5. THE VIDEO DISPLAY BOARD SOFTWARE

VDB4

Introduction

The software can be divided into two parts, namely characters and graphics, although the two types can be mixed and displayed on the screen at the same time.

In the graphics mode the screen is organized as 128 x 48 cells. Entry points are given to access and control the state of each individual cell. Entry points for BASIC program calls are also given in the VDB4 listing in Section V. In the character mode there will be 16 lines of characters each containing 64 characters on the screen. The only customizing normally required is to patch a jump to a "bell" routine to provide the audio if that is required in your system.

In the character mode, two sub modes are allowed, page and scroll; and each allows the option of line foldover. These modes may be read and set by the "U" and "C" commands of AMS-80. They are as follows:

0 page mode with line foldover

1 page mode without line foldover

2 scroll mode with line foldover

3 scroll mode without line foldover

In the scroll sub mode the display scrolls up one line each time a line feed character is output to the display. The screen display will normally start from the top and begin scrolling when the bottom line is reached, i.e. when the screen is full. In the page sub mode, when the screen display is full, a □ (or inverted □ if in the invert video mode) character illuminates in the last character position and the display freezes until a character is depressed at the keyboard. This freeze occurs in any program, even in BASIC because the screen driver software does not return to the calling program until the key is depressed.

If a '/' key was depressed, the screen display will scroll up on line and freeze once more when the screen is full. If a '-' character was depressed, the display software reverts to the scroll mode. If any other character was depressed (the space is the most convenient), the unprotected portion of the screen is erased and characters are written from the top once more. When the screen is full, it freezes up once again. If you use the "holds" code (line 2070 on page C-4), the space bar will give you a new page on the screen, but any other character will be "remembered".

The line foldover feature allows data to hangup at the end of a line, overprinting on the 64th character until a carriage return character is received, or allows the display to continue and automatically advance to the next line to display the 65th and remaining characters in the line: a carriage return character will than return the cursor to the 65th character position and a line feed will advance in the normal way, allowing data contianing more than 64 characters in a line to be displayed in full. The handup feature is useful when tabular data is to be displayed, or when you are not interested in the last few characters on the line.

This video display driver is the most flexible driver yet seen that can live in 512 bytes of memory. Other features include the protection of Ø-3 lines at the top of the screen, control to clear either the entire screen or to clear only the unprotected lines, etc.

```
OBJ             SEQ             SOURCE STATEMENT

                  1 $MACROFILE
                  2 $PAGEWIDTH  (78)
                  3 $PAGELENGTH (60)
                  4 $TITLE  ('AMS80 REV 5.8')
                  5
                  6             NAME    AMS80
                  7
                  8             ;DATE 790313-1
                  9             ;VER 5.8.2 791204
                 10
                 11 FALSE       EQU     0
                 12 TRUE        EQU     NOT FALSE
                 13
                 14 TRUMP       SET     TRUE    ;AMSAT TRUMP CARD
                 15 BYTST       SET     FALSE   ;BYTE STORE CPU
                 16 STPR        SET     FALSE   ;SPACE TIME PROD
                 17 DEBUG       SET     FALSE
                 18
                 19
                 20             ;WORD ASSIGNMENTS
                 21             ;0 1 2 = JUMP TO AMS80
                 22             ;3     = I/O CONFIGURATION
                 23             ;5 6 7 = I/O ROUTINE LOCATION
                 24             ;8 9 A = BREAKPOINT ENTRY
                 25             ;B     = VIDEO CONTROL BYTE (VDB4)
                 26             ;E F   = RAM BLOCK BASE ADDRESS
                 27
                 28
                 29 RS1         EQU     8
                 30 IOCW        EQU     3       ;I/O CONFIGURATION CONTROL
                 31 VDMCW       EQU     0BH     ;VDM MODE CONTOL WORD
                 32 IOEX        EQU     5       ;I/O EXERCISER ROUTINE
                 33 RBADR       EQU     0EH     ;RAM BLOCK BASE ADDRESS
                 34 CASE        EQU     3       ;BACON CASE MEM LOC
                 35             IF      BYTST
                 36 ICP         EQU     0FEH    ;INT PORT
                 37             ENDIF
                 38             IF      TRUMP
                 39 ICP         EQU     0BAH
                 40             ENDIF
                 41
                 42 BASE        EQU     0F000H
                 43
                 44 IOROM       SET     BASE+0C00H
                 45             IF      STPR
                 46 IOROM       SET     0E000H
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

                   47           ENDIF
                   48
                   49
F000               50           ORG     BASE
                   51                   ;INTERFACE AREA
                   52                   ;COMPATABLE TO TDL Z80 MONITOR
F000 C38BFC        53 AMS80:    JMP     CUSTOM  ;START OF SOFTWARE
F003 C3CAF0        54           JMP     CONSI   ;  <A> FROM CONSOLE
F006 C3DFF0        55           JMP     RDR     ;  <A> FROM READER
F009 C38BF0        56           JMP     CONSO   ;  <C> TO CONSOLE
F00C C3A0F0        57           JMP     PUNCH   ;  <C> TO PUNCH
F00F C3B5F0        58           JMP     LIST    ;  <C> TO LIST
F012 C3F4F0        59           JMP     CSTS    ;TEST CONSOLE STATUS
F015 C324F1        60           JMP     IOCHK   ;TEST I/O CONFIGURATION
F018 C328F1        61           JMP     IOSET   ;SET I/O CONFIGURATION
F01B C352F4        62           JMP     MEMCK   ;FIND TOP OF USER AREA
F01E C377F4        63           JMP     RESTART ;BREAKPOINT
                   64
                   65                   ;ENTRY POINTS
F021 C39CF1        66           JMP     START   ;RE-ENTER
F024 C34DF1        67           JMP     BEGIN   ;BYPASS CUSTOMISING AREA
                   68
                   69                   ;CONSOLE ROUTINES
F027 C32DF1        70           JMP     CHIN    ;CONSOLE INPUT AND ECHO
F02A C3DFFB        71           JMP     AOUT    ;<A> TO CONSOLE
F02D C344F9        72           JMP     TCSTS   ;GOTO MON IF CONSOLE INTRPT
F030 C32FF6        73           JMP     TCRET   ;OUTPUT CR/LF
F033 C3A9F5        74           JMP     THXN    ;<A(LSB)> TO CONSOLE
F036 C39BF5        75           JMP     THXB    ;OUTPUT <A>
F039 C304F6        76           JMP     THXW    ;OUTPUT <H/L>
F03C C348F6        77           JMP     MSG     ;OUTPUT TEXT
F03F C342F4        78           JMP     PCHK    ;TEST FOR NULL INPUT CHAR
F042 C309F1        79           JMP     CONSB   ;  <B> TO CONSOLE
                   80
                   81                   ;PUNCH ROUTINES
F045 C380F5        82           JMP     PHXB    ;<A> IN HEX TO PUNCH
F048 C3E6F5        83           JMP     LEAD    ;PUNCH 6 INS LEADER TAPE
F04B C33EF6        84           JMP     PCRET   ;OUTPUT CR/LF
F04E C314F6        85           JMP     PHXW    ;OUTPUT <H/L>
F051 C312F1        86           JMP     POB     ;  <B> TO PUNCH
                   87
                   88
                   89                   ;LIST ROUTINES
F054 C30CF6        90           JMP     LHXW    ;OUTPUT <H/L> TO LIST
F057 C3B1F5        91           JMP     LHXB
F05A C325F6        92           JMP     LCRET
F05D C31BF1        93           JMP     LOB     ;  <B> TO LIST
                   94
                   95                   ;READER ROUTINES
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

0160 C3CFF5         96          JMP     RHXB    ;READ PACKED HEX BYTE
0163 C31CF6         97          JMP     RIX     ;READ CHAR, MASK PARITY BIT
0166 C366F0         98          JMP     $
                    99
                   100                          ;UTILITY ROUTINES
0169 C3C7F5        101          JMP     CONV    ;CONVERT HEX TO ASCII
016C C3F2F5        102          JMP     NIBBLE  ;CONVERT  ASCII TO HEX
016F C338F4        103          JMP     DONE    ;TEST FOR COMPLETION
0172 C35CF9        104          JMP     TIMER   ;DELAY
0175 C3FEF1        105          JMP     SDEHL   ;HL-DE
0178 C348FB        106          JMP     LOCMB   ;LOCATE INDEX IN RAM BLOCK
017B C344FB        107          JMP     IRST    ;RESET INTERRUPTS
017E C3C8FA        108          JMP     BACON   ;BAUDOT TO ASCII CONVERSION
0181 C34DFB        109          JMP     ASCBD   ;ASCII TO BAUDOT CONVERSION
0184 C384F0        110          JMP     $       ;SPACE FOR PATCHES
0187 C387F0        111          JMP     $
                   112
                   113
                   114
                   115
                   116                          ;WORD 3 CONTROLS I/O
                   117          ;-----------------------------------------
                   118          ;LIST :   HI OUT :    HI-IN :     CONSOLE :
                   119          ; 7 6 :    5   4 :     3  2 :       1  0  :
                   120          ;-----------------------------------------
                   121
                   122
                   123 IOST     MACRO   DEV3,DEV2,HB,DEV1,LB,DEV0,MASK
                   124          LDA     IOCW
                   125          ANI     MASK
                   126          JZ      DEV0
                   127          CPI     LB
                   128          JZ      DEV1
                   129          CPI     HB
                   130          JZ      DEV2
                   131          JMP     DEV3
                   132          ENDM
                   133
                   134
                   135 CONSA:   ;SUBROUTINE TO OUTPUT <A> TO CONSOLE
 014F              136          MOV     C,A
                   137 CONSO:   ;SUBROUTINE TO OUTPUT <C> TO CONSOLE
                   138          IOST    CONO3,CONO2,2,CONO1,1,CONO0,3
 3A0300            139+         LDA     IOCW
 E603              140+         ANI     3
 CA2DFC            141+         JZ      CONO0
 FE01              142+         CPI     1
 CA2AFC            143+         JZ      CONO1
 FE02              144+         CPI     2
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

F09A CA27FC        145+         JZ      CONO2
F09D C324FC        146+         JMP     CONO3
                   147 PUNCH:   ;SUBROUTINE  TO OUTPUT <C> TO PUNCH
                   148          IOST    PUN3,PUN2,20H,PUN1,10H,PUN0,30H
F0A0 3A0300        149+         LDA     IOCW
F0A3 E630          150+         ANI     30H
F0A5 CA09FC        151+         JZ      PUN0
F0A8 FE10          152+         CPI     10H
F0AA CA06FC        153+         JZ      PUN1
F0AD FE20          154+         CPI     20H
F0AF CA03FC        155+         JZ      PUN2
F0B2 C300FC        156+         JMP     PUN3
                   157 LIST:    ;SUBROUTINE TO OUTPUT <C> TO LIST DEVICE
                   158          IOST    LIST3,LIST2,80H,LIST1,40H,LIST0,0C0H
F0B5 3A0300        159+         LDA     IOCW
F0B8 E6C0          160+         ANI     0C0H
F0BA CA21FC        161+         JZ      LIST0
F0BD FE40          162+         CPI     40H
F0BF CA1EFC        163+         JZ      LIST1
F0C2 FE80          164+         CPI     80H
F0C4 CA1BFC        165+         JZ      LIST2
F0C7 C318FC        166+         JMP     LIST3
                   167 CONSI:   ;SUBROUTINE TO INPUT  TO <A> FROM CONSOLE
                   168          IOST    CONI3,CONI2,2,CONI1,1,CONI0,3
F0CA 3A0300        169+         LDA     IOCW
F0CD E603          170+         ANI     3
F0CF CA39FC        171+         JZ      CONI0
F0D2 FE01          172+         CPI     1
F0D4 CA36FC        173+         JZ      CONI1
F0D7 FE02          174+         CPI     2
F0D9 CA33FC        175+         JZ      CONI2
F0DC C330FC        176+         JMP     CONI3
                   177 RDR:     ;SUBROUTINE TO INPUT  TO <A> FROM READER
                   178          IOST    RDR3,RDR2,8,RDR1,4,RDR0,0CH
F0DF 3A0300        179+         LDA     IOCW
F0E2 E60C          180+         ANI     0CH
F0E4 CA15FC        181+         JZ      RDR0
F0E7 FE04          182+         CPI     4
F0E9 CA12FC        183+         JZ      RDR1
F0EC FE08          184+         CPI     8
F0EE CA0FFC        185+         JZ      RDR2
F0F1 C30CFC        186+         JMP     RDR3
                   187 CSTS:    ;SUBROUTINE TO TEST CONSOLE STATUS
                   188          IOST    CSTS3,CSTS2,2,CSTS1,1,CSTS0,3
F0F4 3A0300        189+         LDA     IOCW
F0F7 E603          190+         ANI     3
F0F9 CA45FC        191+         JZ      CSTS0
F0FC FE01          192+         CPI     1
F0FE CA42FC        193+         JZ      CSTS1
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

F101 FE02          194+         CPI     2
F103 CA3FFC        195+         JZ      CSTS2
F106 C33CFC        196+         JMP     CSTS3
                   197
                   198
                   199 BOUT     MACRO   DEV
                   200          PUSH    PSW
                   201          PUSH    B
                   202          MOV     C,B      ;SET UP TO
                   203          CALL    DEV      ;OUTPUT  IT
                   204          POP     B
                   205          POP     PSW
                   206          RET
                   207          ENDM
                   208
                   209 CONSB:   ;SUB TO OUTPUT <B> TO CONSOLE
                   210          BOUT    CONSO
F109 F5            211+         PUSH    PSW
F10A C5            212+         PUSH    B
F10B 48            213+         MOV     C,B      ;SET UP TO
F10C CD8BF0        214+         CALL    CONSO    ;OUTPUT  IT
F10F C1            215+         POP     B
F110 F1            216+         POP     PSW
F111 C9            217+         RET
                   218
                   219 POB:     ;SUBROUTINE TO OUTPUT <B> TO PUNCH
                   220          BOUT    PUNCH
F112 F5            221+         PUSH    PSW
F113 C5            222+         PUSH    B
F114 48            223+         MOV     C,B      ;SET UP TO
F115 CDA0F0        224+         CALL    PUNCH    ;OUTPUT  IT
F118 C1            225+         POP     B
F119 F1            226+         POP     PSW
F11A C9            227+         RET
                   228
                   229 LOB:     ;SUBROUTINE TO OUTPUT <B> TO LIST
                   230          BOUT    LIST
F11B F5            231+         PUSH    PSW
F11C C5            232+         PUSH    B
F11D 48            233+         MOV     C,B      ;SET UP TO
F11E CDB5F0        234+         CALL    LIST     ;OUTPUT  IT
F121 C1            235+         POP     B
F122 F1            236+         POP     PSW
F123 C9            237+         RET
                   238
                   239 IOCHK:   ;SUBROUTINE TO GET I/O CONFIGURATION
F124 3A0300        240          LDA     IOCW
F127 C9            241          RET
                   242
```

```
  LOC  OBJ          SEQ          SOURCE STATEMENT

                     243 IOSET:   ;SUBROUTINE TO SET I/O CONFIGURATION
  F128 79            244          MOV     A,C
  F129 320300        245          STA     IOCW
  F12C C9            246          RET
                     247
                     248 CHIN:    ;CONSOLE INPUT AND ECHO ROUTINE
  F12D CDCAF0        249          CALL    CONSI   ;GET CHARACTER
  F130 E67F          250          ANI     7FH     ;MASK PARITY BIT
  F132 C5            251          PUSH    B
  F133 CD8AF0        252          CALL    CONSA   ;OUTPUT IT
  F136 79            253          MOV     A,C     ;RETRIEVE IT
  F137 C1            254          POP     B
  F138 C9            255          RET
                     256
  F139 0C            257 HELLO:   DB      0CH     ;FF TO CLEAR SCREEN
  F13A 414D5338      258          DB      'AMS80'
  F13E 30
  F13F 20            259          DB      ' '
  F140 5245562E      260          DB      'REV. '
  F144 20
  F145 352E382E      261          DB      '5.8.1' ;SPACE TIME VERSION
  F149 31
  F14A 0D            262          DB      0DH,0AH,-1
  F14B 0A
  F14C FF
                     263
                     264
  F14D               265 PROM     SET $
                     266
                     267
  FC00               268 ORG      IOROM
                     269          IF      STPR
                     270          JMP     BASE    ;PATCH FOR BOOT
                     271          ENDIF
                     272 ;INTERFACES TO CUSTOM I/O DRIVERS
                     273 ;SEE USER
                     274
  0003               275 PUN3:    DS      3
  0003               276 PUN2:    DS      3
  0003               277 PUN1:    DS      3
  0003               278 PUN0:    DS      3
                     279
  0003               280 RDR3:    DS      3
  0003               281 RDR2:    DS      3
  0003               282 RDR1:    DS      3
  0003               283 RDR0:    DS      3
                     284
  0003               285 LIST3:   DS      3
  0003               286 LIST2:   DS      3
```

```
LOC  OBJ           SEQ          SOURCE STATEMENT

0003                287 LIST1:  DS      3
0003                288 LIST0:  DS      3
                    289
0003                290 CONO3:  DS      3
0003                291 CONO2:  DS      3
0003                292 CONO1:  DS      3
0003                293 CONO0:  DS      3
                    294
0003                295 CONI3:  DS      3
0003                296 CONI2:  DS      3
0003                297 CONI1:  DS      3
0003                298 CONI0:  DS      3
                    299
0003                300 CSTS3:  DS      3
0003                301 CSTS2:  DS      3
0003                302 CSTS1:  DS      3
0003                303 CSTS0:  DS      3
                    304
0003                305 BTTI:   DS      3
0003                306 BTTO:   DS      3
                    307
0002                308 TBSE:   DS      2
0001                309 IORW:   DS      1       ;MASK FOR I/O CONFIGURATION
                    310
0002                311 TPMR:   DS      2
0002                312 TPMR1:  DS      2
                    313
                    314
                    315
                    316 ;COMMAND TABLE IN CUSTOM AREA, SO PATCHES CAN BE MADE
0036                317 VECTOR: DS      54      ;SPACE FOR COMMAND TABLE
                    318                         ;SEE USER
                    319 CUSTOM:                 ;PUT CUSTOMISING STUFF HERE
FC8B 2A4EFC         320         LHLD    TBSE
FC8E 220E00         321         SHLD    RBADR
FC91 C34DF1         322         JMP     BEGIN
                    323
                    324
F14D                325         ORG     PROM    ;START OF REAL STUFF
                    326
                    327 BEGIN:                  ;PROGRAM ENTRY POINT
F14D 2A51FC         328         LHLD    TPMR    ;FIND END OF MEMORY
F150 25             329 BG0:    DCR     H
F151 7E             330         MOV     A,M     ;FETCH <M>
F152 2F             331         CMA
F153 77             332         MOV     M,A     ;ATTEMPT TO WRITE IT
F154 BE             333         CMP     M       ;DID IT
F155 2F             334         CMA
F156 77             335         MOV     M,A     ;WRITE DATA BACK
```

```
  LOC  OBJ          SEQ          SOURCE STATEMENT

  F157 C250F1        336          JNZ     BG0      ;YES, CONTINUE
  F15A 23            337          INX     H        ;POINT TO FIRST NON RAM LOC
  F15B 0612          338          MVI     B,ENDX-EXIT ;MOVE EXIT TEMPLATE
                     339                          ;TO RAM
  F15D 1141F5        340          LXI     D,ENDX
  F160 1B            341 BG1:     DCX     D
  F161 1A            342          LDAX    D
  F162 2B            343          DCX     H
  F163 77            344          MOV     M,A
  F164 05            345          DCR     B
  F165 C260F1        346          JNZ     BG1
  F168 F9            347          SPHL             ;SET STACK
  F169 01ECFF        348          LXI     B,-20
  F16C 09            349          DAD     B        ;MOVE STACK DOWN FROM TOP
  F16D E5            350          PUSH    H
  F16E 210000        351          LXI     H,0
  F171 E5            352          PUSH    H
  F172 E5            353          PUSH    H
  F173 E5            354          PUSH    H        ;PUSH REGISTERS ON STACK
  F174 3EC3          355          MVI     A,0C3H   ;SET UP FOR JMP RESTART
  F176 320000        356          STA     0
  F179 320800        357          STA     RS1
  F17C 2100F0        358          LXI     H,BASE
  F17F 220100        359          SHLD    1        ;SET UP ENTRY
  F182 2177F4        360          LXI     H,RESTART
  F185 220900        361          SHLD    RS1+1
  F188 3A50FC        362          LDA     IORW     ;GET I/O CONTROL MASK
  F18B 320300        363          STA     IOCW     ;AND SET UP I/O
  F18E 3E02          364          MVI     A,2      ;ENABLE RST 0 &
  F190 CD44FB        365          CALL    IRST     ;RST 1
  F193 CD2FF6        366          CALL    TCRET
  F196 2139F1        367          LXI     H,HELLO
  F199 CD48F6        368          CALL    MSG      ;PRINT OPENING MESSAGE
                     369 START:                    ;MAIN COMMAND LOOP
  F19C CD2FF6        370          CALL    TCRET
  F19F 0E3A          371 STAR1:   MVI     C,':'
  F1A1 CD8BF0        372          CALL    CONSO    ;OUTPUT INITIAL SIGN
  F1A4 CD2DF1        373          CALL    CHIN     ;GET AND ECHO CHARACTER
  F1A7 D640          374          SUI     '@'      ;TEST FOR LETTER
  F1A9 FA9CF1        375          JM      START
  F1AC FE1B          376          CPI     'Z'-'@'+1
  F1AE D29CF1        377          JNC     START    ;INVALID CHARACTER
  F1B1 0E02          378          MVI     C,2      ;SET UP FOR 2 PARAMS
  F1B3 2155FC        379          LXI     H,VECTOR
  F1B6 87            380          ADD     A        ;*2
  F1B7 85            381          ADD     L
  F1B8 6F            382          MOV     L,A
  F1B9 7E            383          MOV     A,M
  F1BA 23            384          INX     H
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

F1BB 66            385          MOV     H,M
F1BC 6F            386          MOV     L,A
F1BD E9            387          PCHL            ;JMP INDIRECT TO IT
                   388
                   389 ERROR:           ;ERROR EXIT
F1BE CD62F4        390          CALL    MEMSIZ  ;COMPUTE TOP OF MEMORY
F1C1 11F8FF        391          LXI     D,-8
F1C4 19            392          DAD     D
F1C5 F9            393          SPHL            ;RESET STACK POINTER
F1C6 CD2FF6        394          CALL    TCRET
F1C9 0E3F          395          MVI     C,'?'
F1CB CD8BF0        396          CALL    CONSO
F1CE C39FF1        397          JMP     STAR1
                   398
                   399
                   400 HEXM:    ;THIS ROUTINE DOES HEX MATH
F1D1 CD04F4        401          CALL    EXPR    ;GET 2 NUMBERS
F1D4 CD2FF6        402          CALL    TCRET   ;OUTPUT CRLF
F1D7 D1            403          POP     D
F1D8 E1            404          POP     H
F1D9 E5            405          PUSH    H       ;SAVE AGAIN
F1DA 19            406          DAD     D       ;COMPUTE SUM
F1DB CD04F6        407          CALL    THXW    ;DISPLAY IT
F1DE D5            408          PUSH    D
F1DF 0E20          409          MVI     C,' '
F1E1 CD8BF0        410          CALL    CONSO
F1E4 D1            411          POP     D
F1E5 E1            412          POP     H
F1E6 E5            413          PUSH    H       ;SAVE ONCE MORE
F1E7 D5            414          PUSH    D
F1E8 CDFEF1        415          CALL    SDEHL   ;SUBTRACT (A-B)
F1EB CD04F6        416          CALL    THXW
F1EE E1            417          POP     H       ;REVERSE ORDER
F1EF D1            418          POP     D
F1F0 CDFEF1        419          CALL    SDEHL   ;(B-A)
F1F3 0E20          420          MVI     C,' '
F1F5 CD8BF0        421          CALL    CONSO
F1F8 CD04F6        422          CALL    THXW
F1FB C39CF1        423          JMP     START
                   424
                   425 SDEHL:   ;ROUTINE TO HL-DE
F1FE 7D            426          MOV     A,L
F1FF 93            427          SUB     E
F200 6F            428          MOV     L,A
F201 7C            429          MOV     A,H
F202 9A            430          SBB     D
F203 67            431          MOV     H,A
F204 C9            432          RET
                   433
```

```
  LOC  OBJ          SEQ          SOURCE STATEMENT

                    434
                    435 MOVE:                       ;MOVE MEMORY
                    436          ;THIS ROUTINE MOVES A BLOCK OF MEMORY
                    437          ;FIRST = START OF BLOCK
                    438          ;SECOND = END OF BLOCK
                    439          ;THIRD = START OF DESTINATION
  F205 0C           440          INR     C          ;SET UP FOR 3
  F206 CD04F4       441          CALL    EXPR       ;GET ADRESSES
  F209 C1           442          POP     B          ;DESTINATION
  F20A D1           443          POP     D          ;SOURCE END
  F20B E1           444          POP     H          ;SOURCE BEGIN
  F20C 7E           445 MV0:     MOV     A,M        ;GET DATA BYTE
  F20D 02           446          STAX    B          ;AND STORE
  F20E 03           447          INX     B          ;INC DESTINATION POINTER
  F20F CD38F4       448          CALL    DONE       ;TEST FOR COMPLETION
  F212 D20CF2       449          JNC     MV0
  F215 C39CF1       450          JMP     START
                    451
                    452
                    453
                    454 FILL:                       ;FILL MEMORY AREA
  F218 0C           455          INR     C          ;GET 3 PARAMETERS
  F219 CD04F4       456          CALL    EXPR
  F21C C1           457          POP     B          ;GET DATA IN <C>
  F21D D1           458          POP     D          ;GET HIGH ADDRESS
  F21E E1           459          POP     H          ;GET LOW ADDRESS
  F21F 71           460 FI0:     MOV     M,C        ;STORE CONSTANT
  F220 CD38F4       461          CALL    DONE       ;TEST FOR COMPLETION
  F223 D21FF2       462          JNC     FI0
  F226 C39CF1       463          JMP     START
                    464
                    465 DISP:                       ;DISPLAY MEMORY
                    466          ;THIS ROUTINE EXPECTS TWO HEX PARAMETERS
                    467          ;LISTING THE BOUNDS OF MEMORY AREA TO BE
                    468          ;DISPLAYED
  F229 CD04F4       469          CALL    EXPR       ;GET TWO ADRESSES
  F22C D1           470          POP     D          ;GET HIGH ADDRESS
  F22D E1           471          POP     H          ;GET LOW ADRESS
  F22E CD2FF6       472 DI0:     CALL    TCRET
  F231 CD04F6       473          CALL    THXW       ;PRINT MEMORY ADRESS
  F234 0E20         474 DI1:     MVI     C,' '
  F236 CD8BF0       475          CALL    CONSO      ;PRINT SPACE
  F239 7E           476          MOV     A,M
  F23A CD9BF5       477          CALL    THXB       ;PRINT DATA
  F23D CD38F4       478          CALL    DONE       ;TEST FOR COMPLETION
  F240 DA9CF1       479          JC      START
  F243 7D           480          MOV     A,L
  F244 E60F         481          ANI     0FH        ;PRINT CR/LF ETC AT LINE E
  F246 C234F2       482          JNZ     DI1
```

```
LOC  OBJ           SEQ          SOURCE STATEMENT

F249 CD44F9         483          CALL    TCSTS   ;TEST KEYBOARD
F24C C32EF2         484          JMP     DI0
                    485
                    486 EOF:                     ;END OF FILE
                    487          ;THIS ROUTINE PRODUCES AN END OF
                    488          ;RECORD ON A PAPER TAPE
                    489
F24F 0D             490          DCR     C       ;GET ONE PARAMETER
F250 CD04F4         491          CALL    EXPR
F253 CD3EF6         492          CALL    PCRET   ;PUNCH CRLF
F256 0E3A           493          MVI     C,':'
F258 CDA0F0         494          CALL    PUNCH
F25B AF             495          XRA     A       ;CLEAR CHECKSUM
F25C 57             496          MOV     D,A
F25D CD80F5         497          CALL    PHXB    ;OUTPUT RECORD LENGTH
F260 E1             498          POP     H
F261 CD14F6         499          CALL    PHXW    ;PUNCH EXECUTION ADRDESS
F264 3E01           500          MVI     A,1     ;RECORD TYPE 1
F266 CD80F5         501          CALL    PHXB
F269 AF             502          XRA     A
F26A 92             503          SUB     D       ;OUTPUT KSUM
F26B CD80F5         504          CALL    PHXB
                    505
                    506 NULL:                    ;NULL THIS ROUTINE PUNCHES
                    507                          ;  60 NULL CHARACTERS
F26E CDE6F5         508          CALL    LEAD
F271 C39CF1         509          JMP     START
                    510
                    511
                    512 READ:                    ;READ ROUTINE
                    513          ;THIS ROUTINE READS A HEX TAPE AND STORES
                    514          ;IT IN MEMORY
F274 0D             515          DCR     C       ;GET ONE ADDRESS
F275 CD04F4         516          CALL    EXPR
F278 E1             517 RED0:    POP     H       ;GET BIAS ADDRESS
F279 E5             518          PUSH    H
F27A CD1CF6         519          CALL    RIX
F27D DE3A           520          SBI     ':'
F27F C278F2         521          JNZ     RED0    ;SCAN TO RECORD MARK
F282 57             522          MOV     D,A     ;CLEAR CHECKSUM
F283 CDCFF5         523          CALL    RHXB
F286 CAA9F2         524          JZ      RED2    ;ZERO RECORD LENGTH,
                    525                          ;ALL DONE
F289 5F             526          MOV     E,A     ;  E <- RECORD LENGTH
F28A CDCFF5         527          CALL    RHXB    ;GET MSB OF LOAD ADDRESS
F28D F5             528          PUSH    PSW     ;SAVE IT
F28E CDCFF5         529          CALL    RHXB    ;GET LSB OF LOAD ADDRESS
F291 C1             530          POP     B       ;RETRIEVE MSB TO <B>
F292 4F             531          MOV     C,A
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

F293 09            532          DAD     B        ;BIAS ADDR + LOAD ADDR -> HL
F294 CDCFF5        533          CALL    RHXB     ;RECORD TYPE
F297 CDCFF5        534 RED1:    CALL    RHXB     ;READ DATA
F29A 77            535          MOV     M,A      ;PUT IN MEMORY
F29B 23            536          INX     H
F29C 1D            537          DCR     E
F29D C297F2        538          JNZ     RED1     ;LOOP UNTIL DONE
F2A0 CDCFF5        539          CALL    RHXB     ;READ CHECKSUM
F2A3 C2BEF1        540          JNZ     ERROR    ;CHECKSUM ERROR
F2A6 C378F2        541          JMP     RED0     ;GET ANOTHER RECORD
F2A9 CDCFF5        542 RED2:    CALL    RHXB     ;GET MSB OF TRANSFER ADDRESS
F2AC 67            543          MOV     H,A
F2AD CDCFF5        544          CALL    RHXB
F2B0 6F            545          MOV     L,A
F2B1 B4            546          ORA     H
F2B2 CAB6F2        547          JZ      RED3     ;IF TRANSFER ADDRESS = 0,
                   548                           ;RETURN TO KB
F2B5 E9            549          PCHL
F2B6 E1            550 RED3:    POP     H
F2B7 C39CF1        551          JMP     START
                   552
                   553
                   554 SUBS:    ;SUBSTITUTE  <MEMORY> ROUTINE
F2BA 0D            555          DCR     C
F2BB CD04F4        556          CALL    EXPR     ;GET ONE ADDRESS
F2BE CD45F4        557          CALL    P2C
F2C1 DABEF1        558          JC      ERROR
F2C4 E1            559          POP     H
F2C5 7E            560 SU0:     MOV     A,M
F2C6 CD9BF5        561          CALL    THXB     ;DISPLAY DATA
F2C9 0E2D          562          MVI     C,'-'
F2CB CD8BF0        563          CALL    CONSO
F2CE CD42F4        564          CALL    PCHK
F2D1 DA9CF1        565          JC      START    ;CR, RETURN TO COMMAND MODE
F2D4 CAE4F2        566          JZ      SU1      ;SPACE ENTERED, SPACE BY
F2D7 E5            567          PUSH    H        ;SAVE   MEMORY ADDRESS
F2D8 CD30F4        568          CALL    EXF      ;GET NEW VALUE
F2DB D1            569          POP     D        ;  <E> = VALUE
F2DC E1            570          POP     H        ;RESTORE MEMORY ADDRESS
F2DD 73            571          MOV     M,E      ;STORE NEW VALUE
F2DE 78            572          MOV     A,B      ;TEST DELIMITER
F2DF FE0D          573          CPI     0DH      ;CR ?
F2E1 CA9CF1        574          JZ      START    ;CR ENTERED AFTER LAST SUB
F2E4 23            575 SU1:     INX     H
F2E5 C3C5F2        576          JMP     SU0
                   577
                   578 WRITE:                    ;WRITE ROUTINE
                   579          ;THIS ROUTINE PUNCHES PAPER TAPE OF MEMORY
                   580          ;AREA SPECIFIED BY PARAMETERS
```

```
LOC  OBJ           SEQ          SOURCE STATEMENT

F2E8 CD04F4         581          CALL    EXPR    ;GET TWO ADDRESSES
F2EB CD2FF6         582          CALL    TCRET
F2EE D1             583          POP     D       ;GET HIGH ADRESS
F2EF E1             584          POP     H       ;GET LOW ADDRESS
F2F0 7D             585 WRI0:    MOV     A,L
F2F1 C610           586          ADI     16
F2F3 4F             587          MOV     C,A
F2F4 7C             588          MOV     A,H
F2F5 CE00           589          ACI     0
F2F7 47             590          MOV     B,A
F2F8 7B             591          MOV     A,E
F2F9 91             592          SUB     C
F2FA 4F             593          MOV     C,A
F2FB 7A             594          MOV     A,D
F2FC 98             595          SBB     B
F2FD DA05F3         596          JC      WRI1    ;RECORD LENGTH = 16
F300 3E10           597          MVI     A,16
F302 C308F3         598          JMP     WRI2
F305 79             599 WRI1:    MOV     A,C     ;LAST RECORD
F306 C611           600          ADI     17
F308 B7             601 WRI2:    ORA     A
F309 CA9CF1         602          JZ      START
F30C D5             603          PUSH    D       ;SAVE HIGH ADDRESS
F30D 5F             604          MOV     E,A     ; <E> = LENGTH
F30E 1600           605          MVI     D,0     ;CLEAR CHECKSUM
F310 CD3EF6         606          CALL    PCRET   ;PUNCH CRLF
F313 0E3A           607          MVI     C,':'
F315 CDA0F0         608          CALL    PUNCH
F318 7B             609          MOV     A,E
F319 CD80F5         610          CALL    PHXB    ;PUNCH LENGTH
F31C CD14F6         611          CALL    PHXW    ;PUNCH ADDRESS
F31F AF             612          XRA     A
F320 CD80F5         613          CALL    PHXB    ;PUNCH RECORD TYPE
F323 7E             614 WRI3:    MOV     A,M
F324 23             615          INX     H
F325 CD80F5         616          CALL    PHXB    ;PUNCH DATA
F328 1D             617          DCR     E       ;DECREMENT LENGTH
F329 C323F3         618          JNZ     WRI3    ;CONTINUE LOOPING
F32C AF             619          XRA     A
F32D 92             620          SUB     D
F32E CD80F5         621          CALL    PHXB    ;PUNCH CHECKSUM
F331 D1             622          POP     D       ;RESTORE HIGH ADDRESS
F332 C3F0F2         623          JMP     WRI0
                    624
                    625
                    626 ASCI:                    ;DUMP <MEMORY> IN ASCII
F335 CD04F4         627          CALL    EXPR
F338 D1             628          POP     D       ;GET HI ADDRESS
F339 E1             629          POP     H       ;GET LOW ADDRESS
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

F33A CD44F9        630 ASC3:   CALL    TCSTS
F33D CD2FF6        631         CALL    TCRET
F340 CD04F6        632         CALL    THXW     ;PRINT ADDRESS
F343 0E20          633         MVI     C,' '
F345 CD8BF0        634         CALL    CONSO
F348 7E            635 ASC2:   MOV     A,M      ;FETCH BYTE
F349 FEFF          636         CPI     -1
F34B CC65F3        637         CZ      ASC1     ;FUDGE "RUBOUT"
F34E FE20          638         CPI     ' '
F350 FC65F3        639         CM      ASC1
F353 CD8AF0        640         CALL    CONSA
F356 CD38F4        641         CALL    DONE     ;TEST FOR COMPLETION
F359 DA9CF1        642         JC      START
F35C 7D            643         MOV     A,L
F35D E60F          644         ANI     0FH
F35F C248F3        645         JNZ     ASC2     ;PRINT CR/LF <H/L> ETG
F362 C33AF3        646         JMP     ASC3
F365 3E20          647 ASC1:   MVI     A,' '
F367 C9            648         RET
                   649
                   650
                   651 X:              ;EXAMINE
                   652                 ;THIS ROUTINE ALLOWS REGISTERS
                   653                 ;  TO BE EXAMINED/MODIFED
F368 CD2DF1        654         CALL    CHIN     ;GET REGISTER IDENTIFIER
F36B 2141F5        655         LXI     H,ACTBL  ;POINT TO ACCESS TABLE
F36E FE0D          656         CPI     0DH      ;CR
F370 CAD3F3        657         JZ      X6       ;FULL REGISTER DISPLAY
F373 BE            658 X0:     CMP     M
F374 CA84F3        659         JZ      X1
F377 F5            660         PUSH    PSW      ;SAVE THE CHARACTER
F378 7E            661         MOV     A,M
F379 B7            662         ORA     A        ;TEST FOR END OF TABLE
F37A FABEF1        663         JM      ERROR
F37D 23            664         INX     H
F37E 23            665         INX     H
F37F 23            666         INX     H
F380 F1            667         POP     PSW
F381 C373F3        668         JMP     X0
F384 0E20          669 X1:     MVI     C,' '
F386 CD8BF0        670         CALL    CONSO
F389 23            671 X2:     INX     H
F38A 7E            672         MOV     A,M      ;DISPLACEMENT
F38B EB            673         XCHG             ;SAVE <M> IN <DE>
                   674                          ;  <M> POINTER TO ACTBL
F38C 6F            675         MOV     L,A
F38D 2600          676         MVI     H,0
F38F 39            677         DAD     SP
F390 EB            678         XCHG             ;RESTORE HL
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

F391 23           679          INX     H
F392 46           680          MOV     B,M     ;NUMBER OF BYTES
F393 23           681          INX     H       ;POINT AT NEXT REG IDENT
F394 1A           682          LDAX    D       ;8/16 BIT DISPLAY AND MOD
F395 CD9BF5       683          CALL    THXB    ;MSB OF 16 BIT REG,
                  684                          ;ALL OF 8 BIT REG
F398 05           685          DCR     B
F399 CAA1F3       686          JZ      X3      ;8 BIT DISPLAY
F39C 1B           687          DCX     D
F39D 1A           688          LDAX    D
F39E CD9BF5       689          CALL    THXB    ;LSB OF 16 BIT REG
F3A1 04           690 X3:      INR     B
F3A2 0E2D         691          MVI     C,'-'
F3A4 CD8BF0       692          CALL    CONSO
F3A7 CD42F4       693          CALL    PCHK    ;SKIP IF NULL ENTRY
F3AA DA9CF1       694          JC      START
F3AD CAC5F3       695          JZ      X5
F3B0 E5           696          PUSH    H       ;SAVE POINTER TO ACTBL
F3B1 C5           697          PUSH    B       ;SAVE NUMBER OF BYTES
F3B2 CD30F4       698          CALL    EXF     ;GET NEW REG VALUE
F3B5 E1           699          POP     H
F3B6 F1           700          POP     PSW     ;  <A> = NUMBER OF BYTES
F3B7 C5           701          PUSH    B       ;  <B> = DELIMITER CHAR
F3B8 F5           702          PUSH    PSW     ;  <A> = NUMBER OF BYTES
F3B9 7D           703          MOV     A,L
F3BA 12           704          STAX    D       ;STORE LSB IN REGISTER AREA
F3BB C1           705          POP     B       ;RETRIEVE NUMBER OF BYTES
F3BC 05           706          DCR     B
F3BD CAC3F3       707          JZ      X4      ;8 BITS ONLY
F3C0 13           708          INX     D
F3C1 7C           709          MOV     A,H
F3C2 12           710          STAX    D       ;STORE MSB IN REGISTER AREA
F3C3 C1           711 X4:      POP     B       ;RETRIEVE DELIMITER CHAR
F3C4 E1           712          POP     H
F3C5 7E           713 X5:      MOV     A,M     ;TEST FOR END OF TABLE
F3C6 B7           714          ORA     A
F3C7 FA9CF1       715          JM      START
F3CA 78           716          MOV     A,B     ;TEST DELIMITER
F3CB FE0D         717          CPI     0DH     ;CR
F3CD CA9CF1       718          JZ      START
F3D0 C389F3       719          JMP     X2
F3D3 CD2FF6       720 X6:      CALL    TCRET   ;FULL REGISTER DISPLAY
F3D6 0E20         721 X7:      MVI     C,' '
F3D8 CD8BF0       722          CALL    CONSO   ;OUTPUT A SPACE
F3DB 7E           723          MOV     A,M     ;GET THE CHARACTER
F3DC 23           724          INX     H       ;POINT TO NUMBER OF BYTES
F3DD B7           725          ORA     A       ;SET CONDITION CODES
F3DE FA9CF1       726          JM      START   ;ALL DONE
F3E1 4F           727          MOV     C,A
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

F3E2 CD8BF0        728          CALL    CONSO   ;PRINT CHARACTER
F3E5 0E3D          729          MVI     C,'='
F3E7 CD8BF0        730          CALL    CONSO   ;PRINT = SIGN
F3EA 7E            731          MOV     A,M     ;FETCH NUMBER OF BYTES
F3EB 23            732          INX     H       ;POINT AT NUMBER OF BYTES
F3EC EB            733          XCHG            ;SAVE <HL> IN <DE>
F3ED 6F            734          MOV     L,A
F3EE 2600          735          MVI     H,0
F3F0 39            736          DAD     SP      ;COMPUTE ADDRESS OF VALUE
F3F1 EB            737          XCHG            ;  <HL> = CURRENT TABLE ADDR
                   738                          ;  <DE> = MEMORY ADDR
F3F2 46            739          MOV     B,M     ;FETCH NUMBER OF BYTES
F3F3 23            740          INX     H       ;POINT TO NEXT DISPLAY
F3F4 1A            741          LDAX    D       ;GET LSB OF DATA
F3F5 CD9BF5        742          CALL    THXB    ;DISPLAY IT
F3F8 05            743          DCR     B
F3F9 CAD6F3        744          JZ      X7      ;ALL DONE
F3FC 1B            745          DCX     D       ;POINT TO MSB OF 16 BIT VAL
F3FD 1A            746          LDAX    D       ;FETCH IT
F3FE CD9BF5        747          CALL    THXB    ;DISPLAY IT
F401 C3D6F3        748          JMP     X7
                   749
                   750
                   751 EXPR:    ;EVALUATE EXPRESSION:<1>,<2>,<3>
                   752          ;<C> = NUMBER OF PARAMETERS REQD
                   753          ;PARAMETERS ARE RETURNED ON THE STACK
F404 210000        754          LXI     H,0     ;INITIAL VALUE OF PARAMETER
F407 CDE5FB        755 EX0:     CALL    CHINS   ;GET A CHARACTER
                   756                          ;(IGNORE SPACES)
F40A 47            757 EX1:     MOV     B,A     ;SAVE DELIMITER CHARACTER
F40B CDF2F5        758          CALL    NIBBLE  ;CONVERT TO HEX
F40E DA1AF4        759          JC      EX2     ;NOT LEGAL CHARACTER
F411 29            760          DAD     H       ;*2
F412 29            761          DAD     H       ;*4
F413 29            762          DAD     H       ;*8
F414 29            763          DAD     H       ;*16
F415 B5            764          ORA     L
F416 6F            765          MOV     L,A
F417 C307F4        766          JMP     EX0     ;GET ANOTHER CHARACTER
F41A E3            767 EX2:     XTHL            ;GET RET ADDR OF STACK
                   768                          ;PUT HL ON
F41B E5            769          PUSH    H
F41C 78            770          MOV     A,B
F41D CD45F4        771          CALL    P2C     ;TEST DELIMITER CHAR
F420 D228F4        772          JNC     EX3
F423 0D            773          DCR     C       ;CR ENTERED
F424 C2BEF1        774          JNZ     ERROR   ;TOO FEW PARAMS
F427 C9            775          RET
F428 C2BEF1        776 EX3:     JNZ     ERROR   ;ILLEGAL DELIMITER
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

F42B 0D           777          DCR     C
F42C C204F4       778          JNZ     EXPR
F42F C9           779          RET
F430 0E01         780 EXF:     MVI     C,1        ;ENTRY FOR COND PARAMS
F432 210000       781          LXI     H,0
F435 C30AF4       782          JMP     EX1
                  783
                  784 DONE:                       ;COMPARE <M> WITH <DE>
                  785                             ;C=0, IF <HL> <= <DE>
F438 23           786          INX     H          ;BUMP   HL
F439 7C           787          MOV     A,H        ;TEST FOR <HL> = 0
F43A B5           788          ORA     L
F43B 37           789          STC
F43C C8           790          RZ
F43D 7B           791          MOV     A,E        ;<DE> - <HL> S/R C
F43E 95           792          SUB     L
F43F 7A           793          MOV     A,D
F440 9C           794          SBB     H
F441 C9           795          RET
                  796
                  797
                  798 PCHK:                       ;TEST FOR NULL INPUT PARAM
F442 CD2DF1       799          CALL    CHIN       ;GET A CHARACTER
F445 FE20         800 P2C:     CPI     ' '
F447 C8           801          RZ
F448 FE2C         802          CPI     ','
F44A C8           803          RZ
F44B FE0D         804          CPI     0DH        ;CR
F44D 37           805          STC
F44E 3F           806          CMC                ;CLEAR CARRY FLAG
F44F C0           807          RNZ
F450 37           808          STC
F451 C9           809          RET
                  810
                  811
                  812 MEMCK:                      ;EXTERNALLY REFERENCED ROUTINE
                  813                             ;RETURNS TOP OF USER RAM
                  814                             ;VALUE RETURNED IN (B,A)
F452 E5           815          PUSH    H
F453 CD5BF4       816          CALL    MEMSAZ
F456 44           817          MOV     B,H
F457 3EC0         818          MVI     A,0C0H
F459 E1           819          POP     H
F45A C9           820          RET
                  821
                  822 MEMSAZ: ;STOPS RAM AT 0FFFFH FROM BEING FOUND
F45B C5           823          PUSH    B
F45C 2A53FC       824          LHLD    TPMR1
F45F C366F4       825          JMP     MEM0
```

```
 LOC  OBJ          SEQ          SOURCE STATEMENT

                    826
                    827
                    828 MEMSIZ: ;FIND END OF MEMORY, SET STACK
F462 C5             829         PUSH    B       ;SAVE <BC>
F463 2A51FC         830         LHLD    TPMR    ;FIND END OF MEMORY
F466 25             831 MEM0:   DCR     H
F467 7E             832         MOV     A,M     ;FETCH <M>
F468 2F             833         CMA
F469 77             834         MOV     M,A
F46A BE             835         CMP     M
F46B 2F             836         CMA
F46C 77             837         MOV     M,A
F46D C266F4         838         JNZ     MEM0
F470 23             839         INX     H
F471 01EEFF         840         LXI     B,EXIT-ENDX     ;COMPUTE NEW TOP
F474 09             841         DAD     B
F475 C1             842         POP     B       ;RESTORE <BC>
F476 C9             843         RET
                    844
                    845 RESTART:;THIS ROUTINE SAVES CALLING STATE
                    846         ;AND RETURNS CONTROL TO MONITOR
F477 E5             847         PUSH    H       ;SAVE STATE
F478 D5             848         PUSH    D
F479 C5             849         PUSH    B
F47A F5             850         PUSH    PSW
F47B CD62F4         851         CALL    MEMSIZ  ; <M> NEW STACK POINTER
F47E EB             852         XCHG
F47F 210A00         853         LXI     H,10
F482 39             854         DAD     SP      ;COMPUTE ORIG SP IN STACK
F483 0604           855         MVI     B,4     ;MOVE THE STACK
F485 EB             856         XCHG
F486 2B             857 RST0:   DCX     H
F487 72             858         MOV     M,D
F488 2B             859         DCX     H
F489 73             860         MOV     M,E
F48A D1             861         POP     D
F48B 05             862         DCR     B
F48C C286F4         863         JNZ     RST0
F48F C1             864         POP     B       ;OLD PC = BC, OLD HL = DE
F490 0B             865         DCX     B       ;POINT AT TRAPPED CODE
F491 F9             866         SPHL            ;NEW STACK VALUE
F492 211400         867         LXI     H,TSAV  ;IN THE STACK
F495 39             868         DAD     SP
F496 7E             869         MOV     A,M     ;TEST FOR PROG RESTART
F497 91             870         SUB     C       ;OR CONSOLE
F498 23             871         INX     H
F499 C2A1F4         872         JNZ     RST1
F49C 7E             873         MOV     A,M
F49D 90             874         SUB     B
```

```
LOC  OBJ           SEQ          SOURCE STATEMENT

F49E CAAFF4         875          JZ      RST3
F4A1 23             876 RST1:    INX     H
F4A2 23             877          INX     H
F4A3 7E             878          MOV     A,M
F4A4 91             879          SUB     C
F4A5 C2AEF4         880          JNZ     RST2
F4A8 23             881          INX     H
F4A9 7E             882          MOV     A,M
F4AA 90             883          SUB     B
F4AB CAAFF4         884          JZ      RST3
F4AE 03             885 RST2:    INX     B
F4AF 210F00         886 RST3:    LXI     H,LSAV
F4B2 39             887          DAD     SP
F4B3 73             888          MOV     M,E
F4B4 23             889          INX     H
F4B5 72             890          MOV     M,D      ;SAVE OLD HL
F4B6 23             891          INX     H
F4B7 23             892          INX     H
F4B8 71             893          MOV     M,C      ;SAVE OLD PC
F4B9 23             894          INX     H
F4BA 70             895          MOV     M,B
F4BB C5             896          PUSH    B
F4BC CD2FF6         897          CALL    TCRET
F4BF 0E25           898          MVI     C,'%'
F4C1 CD8BF0         899          CALL    CONSO
F4C4 E1             900          POP     H        ;RETRIEVE OLD PC FOR DISPLAY
F4C5 CD04F6         901          CALL    THXW     ;DISPLAY IT
F4C8 211400         902          LXI     H,TSAV   ;CLEAR TRAPS
F4CB 39             903          DAD     SP       ;IN THE STACK
F4CC 1602           904          MVI     D,2      ;SET UP FOR TWO TRAPS
F4CE 4E             905 RST4:    MOV     C,M      ;GET LSB OF ADDRESS
F4CF 3600           906          MVI     M,0      ;CLEAR MEMORY
F4D1 23             907          INX     H
F4D2 46             908          MOV     B,M      ;GET MSB OF ADDRESS
F4D3 3600           909          MVI     M,0
F4D5 23             910          INX     H
F4D6 79             911          MOV     A,C
F4D7 B0             912          ORA     B
F4D8 CADDF4         913          JZ      RST5     ;TEST FOR VALID TRAP
F4DB 7E             914          MOV     A,M      ;GET OPCODE BYTE
F4DC 02             915          STAX    B        ;REPLACE TI
F4DD 23             916 RST5:    INX     H        ;POINT TO NEXT TRAP ADDERSS
F4DE 15             917          DCR     D
F4DF C2CEF4         918          JNZ     RST4     ;REPEAT FOR TRAP 2
F4E2 C39CF1         919          JMP     START
                    920
                    921
                    922 GOTO:                     ;TRANSFER CONTROL SUBROUTINE
F4E5 CD42F4         923          CALL    PCHK     ;GET A CHARACTER
```

```
 LOC  OBJ          SEQ          SOURCE STATEMENT

 F4E8 DA27F5        924          JC      GO3      ;CR, EXIT
 F4EB CAFFF4        925          JZ      GO0      ;DON'T MODIFY PC
 F4EE CD30F4        926          CALL    EXF      ;GET NEW PC VALUE
 F4F1 D1            927          POP     D
 F4F2 211300        928          LXI     H, PSAV
 F4F5 39            929          DAD     SP
 F4F6 72            930          MOV     M, D     ;STORE MODIFIED PC IN RAM
 F4F7 2B            931          DCX     H
 F4F8 73            932          MOV     M, E
 F4F9 78            933          MOV     A, B     ;RETRIEVE DELIMITER CHARAC
 F4FA FE0D          934          CPI     0DH      ;CR
 F4FC CA27F5        935          JZ      GO3      ;NO TRAPS TO BE SET
 F4FF 1602          936 GO0:     MVI     D, 2     ;SET MAX OF 2 TRAPS
 F501 211400        937          LXI     H, TSAV  ;IN THE STACK
 F504 39            938          DAD     SP
 F505 E5            939 GO1:     PUSH    H        ;SAVE ADDRESS OF TRAP AREA
 F506 0E01          940          MVI     C, 1
 F508 CD04F4        941          CALL    EXPR     ;GET A TRAP ADDRESS
 F50B 58            942          MOV     E, B     ;SAVE DELIMITER CHARACTER
 F50C C1            943          POP     B        ;GET ADDRESS IN BC
 F50D E1            944          POP     H
 F50E 78            945          MOV     A, B
 F50F B1            946          ORA     C
 F510 CA1DF5        947          JZ      GO2      ;DON'T ALLOW A TRAP AT 0
 F513 71            948          MOV     M, C     ;PUT TRAP ADDRESS AWAY
 F514 23            949          INX     H
 F515 70            950          MOV     M, B
 F516 23            951          INX     H
 F517 0A            952          LDAX    B        ;FETCH OPCODE
 F518 77            953          MOV     M, A     ;PUT IN TRAP AREA
 F519 23            954          INX     H
 F51A 3ECF          955          MVI     A, 0CFH  ;RESTART 1
 F51C 02            956          STAX    B        ;SET TRAP IN MEMORY
 F51D 7B            957 GO2:     MOV     A, E     ;TEST DELIMITER CHARACTER
 F51E FE0D          958          CPI     0DH      ;CR
 F520 CA27F5        959          JZ      GO3      ;ALL    DONE
 F523 15            960          DCR     D
 F524 C205F5        961          JNZ     GO1      ;GO TO NEXT TRAP
 F527 CD2FF6        962 GO3:     CALL    TCRET
 F52A 210800        963          LXI     H, 8
 F52D 39            964          DAD     SP
 F52E E9            965          PCHL             ;TAKE BRANCH
                    966
                    967 EXIT:                     ;MONITOR STACK ORIGIN
 F52F D1            968          POP     D
 F530 C1            969          POP     B
 F531 F1            970          POP     PSW
 F532 E1            971          POP     H
 F533 F9            972          SPHL
```

```
LOC  OBJ           SEQ          SOURCE STATEMENT

F534 00             973          NOP
F535 210000         974          LXI     H,$-$
F536                975 HLX      EQU     $-2
F538 C30000         976          JMP     $-$     ;RETURN TO INTERRUPTED CODE
F539                977 PCX      EQU     $-2
F53B 0000           978 T1A:     DW      0       ;TRAP 1 ADDRESS
F53D 00             979          DB      0       ;       VALUE
F53E 0000           980          DW      0       ;TRAP 2 ADDRESS
F540 00             981          DB      0       ;       VALUE
                    982
                    983 ENDX:            ;DISP OF REG SAV FROM SP
0005                984 ASAV     EQU     5
0003                985 BSAV     EQU     3
0002                986 CSAV     EQU     2
0001                987 DSAV     EQU     1
0000                988 ESAV     EQU     0
0004                989 FSAV     EQU     4
0010                990 HSAV     EQU     HLX-EXIT+9
000F                991 LSAV     EQU     HLX-EXIT+8
0013                992 PSAV     EQU     PCX-EXIT+9
0007                993 SSAV     EQU     7
0014                994 TSAV     EQU     T1A-EXIT+8
                    995 ACTBL:           ;TABLE FOR ACCESSING REGISTERS
                    996                  ;TABLE CONTAINING
                    997                  ; 1 REG ID, 2 SP DISP, 3 # OF BYTES
F541 41             998          DB      'A',ASAV,1
F542 05
F543 01
F544 42             999          DB      'B',BSAV,1
F545 03
F546 01
F547 43            1000          DB      'C',CSAV,1
F548 02
F549 01
F54A 44            1001          DB      'D',DSAV,1
F54B 01
F54C 01
F54D 45            1002          DB      'E',ESAV,1
F54E 00
F54F 01
F550 46            1003          DB      'F',FSAV,1
F551 04
F552 01
F553 48            1004          DB      'H',HSAV,1
F554 10
F555 01
F556 4C            1005          DB      'L',LSAV,1
F557 0F
F558 01
```

```
 LOC  OBJ          SEQ          SOURCE STATEMENT

F559 4D           1006          DB      'M',HSAV,2
F55A 10
F55B 02
F55C 50           1007          DB      'P',PSAV,2
F55D 13
F55E 02
F55F 53           1008          DB      'S',SSAV,2
F560 07
F561 02
F562 FF           1009          DB      -1
                  1010
                  1011
                  1012
                  1013
                  1014 HEAD:    ;SUB TO TITLE TAPE
F563 CD2FF6       1015          CALL    TCRET   ;NEW LINE
F566 0E3D         1016          MVI     C,'='
F568 CD8BF0       1017          CALL    CONSO   ;PROMPT
F56B CD2DF1       1018 HEAD1:   CALL    CHIN
F56E FE0D         1019          CPI     0DH     ;TEST FOR CRET
F570 CA7AF5       1020          JZ      HEAD2   ;EXIT
F573 4F           1021          MOV     C,A     ;SET UP TO
F574 CDA0F0       1022          CALL    PUNCH   ;PUNCH CHARAC
F577 C36BF5       1023          JMP     HEAD1   ;NEXT
F57A CD3EF6       1024 HEAD2:   CALL    PCRET
F57D C39CF1       1025          JMP     START
                  1026
                  1027
                  1028          ;USER UTILITY ROUTINES
                  1029 PHXB:    ;PUNCH PACKED BYTE
F580 F5           1030          PUSH    PSW
F581 0F           1031          RRC
F582 0F           1032          RRC
F583 0F           1033          RRC
F584 0F           1034          RRC
F585 E60F         1035          ANI     0FH
F587 CDC7F5       1036          CALL    CONV
F58A CDA0F0       1037          CALL    PUNCH
F58D F1           1038          POP     PSW
F58E F5           1039          PUSH    PSW
F58F E60F         1040          ANI     0FH
F591 CDC7F5       1041          CALL    CONV
F594 CDA0F0       1042          CALL    PUNCH
F597 F1           1043          POP     PSW
F598 82           1044          ADD     D
F599 57           1045          MOV     D,A
F59A C9           1046          RET
                  1047
                  1048 THXB:              ;OUTPUT <A> TO CONSOLE
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

F59B F5           1049          PUSH    PSW     ;SAVE BYTE
F59C 0F           1050          RRC
F59D 0F           1051          RRC
F59E 0F           1052          RRC
F59F 0F           1053          RRC
F5A0 E60F         1054          ANI     0FH     ;MASK LSB
F5A2 CDC7F5       1055          CALL    CONV    ;CONVERT BCD TO ASCII
F5A5 CD8BF0       1056          CALL    CONSO   ;AND OUTPUT IT
F5A8 F1           1057          POP     PSW     ;RETRIEVE BYTE
                  1058 THXN:                    ;OUTPUT LOWER BYTE
F5A9 E60F         1059          ANI     0FH     ;AND MASK IT
F5AB CDC7F5       1060          CALL    CONV
F5AE C38BF0       1061          JMP     CONSO
                  1062
                  1063 LHXB:                    ;LIST PACKED BYTE
F5B1 F5           1064          PUSH    PSW
F5B2 0F           1065          RRC
F5B3 0F           1066          RRC
F5B4 0F           1067          RRC
F5B5 0F           1068          RRC
F5B6 E60F         1069          ANI     0FH
F5B8 CDC7F5       1070          CALL    CONV
F5BB CDB5F0       1071          CALL    LIST
F5BE F1           1072          POP     PSW
F5BF E60F         1073 LHXN:    ANI     0FH
F5C1 CDC7F5       1074          CALL    CONV
F5C4 C3B5F0       1075          JMP     LIST
                  1076
                  1077
                  1078 CONV:                    ;CONVERT HEX TO ASCII
F5C7 C690         1079          ADI     90H     ;SET UP <A> SO THAT
F5C9 27           1080          DAA             ;A THRU F SET CARRY
F5CA CE40         1081          ACI     40H     ;ADD IN CARRY AND
F5CC 27           1082          DAA             ;ADJUST UPPER NIBBLE
F5CD 4F           1083          MOV     C,A
F5CE C9           1084          RET
                  1085
                  1086
                  1087 RHXB:    ;ROUTINE TO READ TWO ASCII CHARS
                  1088                  ;AND DECODE TO 8 BITS BINARY
F5CF CD1CF6       1089          CALL    RIX     ;READ FROM TAPE
F5D2 CDF2F5       1090          CALL    NIBBLE  ;CONVERT TO ASCII
F5D5 07           1091          RLC
F5D6 07           1092          RLC
F5D7 07           1093          RLC
F5D8 07           1094          RLC
F5D9 4F           1095          MOV     C,A     ;AND SAVE
F5DA CD1CF6       1096          CALL    RIX     ;NEXT BYTE
F5DD CDF2F5       1097          CALL    NIBBLE
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

F5E0 B1           1098         ORA     C
F5E1 4F           1099         MOV     C,A
F5E2 82           1100         ADD     D          ;UPDATE CHECKSUM
F5E3 57           1101         MOV     D,A
F5E4 79           1102         MOV     A,C
F5E5 C9           1103         RET
                  1104
                  1105
                  1106 LEAD:                      ;SUBROUTINE TO PUNCH NULLS
F5E6 063C         1107         MVI     B,60       ;SET UP FOR 6 INCHES
F5E8 0E00         1108 LE0:    MVI     C,0
F5EA CDA0F0       1109         CALL    PUNCH
F5ED 05           1110         DCR     B
F5EE C2E8F5       1111         JNZ     LE0
F5F1 C9           1112         RET
                  1113
                  1114
                  1115 NIBBLE:            ;DECODE ASCII CHARAC IN <A>
                  1116                    ;INTO HEX DIGIT IN <A>, ALLOW ONLY
                  1117                    ;CHARS 0..F, STC FOR ILLEGAL CHAR
F5F2 D630         1118         SUI     '0'
F5F4 D8           1119         RC                 ;FILTER 0-2FH
F5F5 C6E9         1120         ADI     '0'-'G'
F5F7 D8           1121         RC                 ;FILTER OUT 47H-0FFH
F5F8 C606         1122         ADI     6
F5FA F200F6       1123         JP      NI0        ;BRANCH A-F
F5FD C607         1124         ADI     7
F5FF D8           1125         RC                 ;FILTER OUT 3AH-40H
F600 C60A         1126 NI0:    ADI     10
F602 B7           1127         ORA     A          ;CLEAR ERROR FLAG
F603 C9           1128         RET
                  1129
                  1130 THXW:                      ;OUTPUT <H/L> TO CONSOLE
F604 7C           1131         MOV     A,H
F605 CD9BF5       1132         CALL    THXB
F608 7D           1133         MOV     A,L
F609 C39BF5       1134         JMP     THXB
                  1135
                  1136 LHXW:                      ;LIST <HL> IN HEX
F60C 7C           1137         MOV     A,H
F60D CDB1F5       1138         CALL    LHXB
F610 7D           1139         MOV     A,L
F611 C3B1F5       1140         JMP     LHXB
                  1141
                  1142 PHXW:                      ;PUNCH <HL> IN HEX
F614 7C           1143         MOV     A,H
F615 CD80F5       1144         CALL    PHXB
F618 7D           1145         MOV     A,L
F619 C380F5       1146         JMP     PHXB
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

                  1147
                  1148 RIX:                        ;READ CHAR, MASK PARITY BIT
F61C CDDFF0       1149          CALL    RDR
F61F DABEF1       1150          JC      ERROR      ;READER TIMEOUT ERROR
F622 E67F         1151          ANI     7FH
F624 C9           1152          RET
                  1153
                  1154
                  1155 LCRET:                      ;SUBROUTINE
F625 0E0D         1156          MVI     C,0DH
F627 CD85F0       1157          CALL    LIST
F62A 0E0A         1158          MVI     C,0AH
F62C C385F0       1159          JMP     LIST
                  1160
                  1161
                  1162 TCRET:   ;SUB TO OUTPUT CRLF TO CONSOLE
F62F 0E0D         1163          MVI     C,0DH
F631 CD8BF0       1164          CALL    CONSO
F634 0E0D         1165          MVI     C,0DH
F636 CD8BF0       1166          CALL    CONSO
F639 0E0A         1167          MVI     C,0AH
F63B C38BF0       1168          JMP     CONSO
                  1169
                  1170 PCRET:                      ;SUBROUTINE TO PUNCH CRLF
F63E 0E0D         1171          MVI     C,0DH
F640 CDA0F0       1172          CALL    PUNCH
F643 0E0A         1173          MVI     C,0AH
F645 C3A0F0       1174          JMP     PUNCH
                  1175
                  1176
                  1177 ;SUBROUTINE TO PRINT MESSAGE, FORMAT N,ASCII
                  1178          ;<M> POINTS TO FIRST BYTE
                  1179
                  1180 MSG:                        ;SUBROUTINE TO PRINT
                  1181                             ; <M> POINTS TO FIRST BYTE
                  1182                             ;MESSAGE , FORMAT  'ASCII',0FFH
F648 7E           1183          MOV     A,M        ;FETCH
F649 FEFF         1184          CPI     -1         ;TEST FOR END OF MESSAGE
F64B C8           1185          RZ                 ;OK, EXIT
F64C CD8AF0       1186          CALL    CONSA      ;PRINT IT
F64F 23           1187          INX     H          ;POINT TO NEXT CHARACTER
F650 C348F6       1188          JMP     MSG
                  1189
                  1190
                  1191          ;INSPECT SUBROUTINE
                  1192 ;ENTRY POINT "INSPT"
F653 E1           1193 RDBYTE:  POP     H
F654 E3           1194          XTHL
F655 7E           1195          MOV     A,M
```

```
  LOC  OBJ          SEQ          SOURCE STATEMENT

  F656 23           1196           INX     H
  F657 E3           1197           XTHL
  F658 E5           1198           PUSH    H
  F659 C9           1199           RET
                    1200
                    1201
  F65A 3C           1202 RGPRNT:   INR     A
  F65B E607         1203           ANI     7
  F65D FE06         1204           CPI     6
  F65F DA64F6       1205           JC      RGP1
  F662 C603         1206           ADI     3
  F664 FE05         1207 RGP1:     CPI     5
  F666 DA6BF6       1208           JC      RGP2
  F669 C602         1209           ADI     2
  F66B C641         1210 RGP2:     ADI     41H
  F66D 4F           1211           MOV     C,A
  F66E C38BF0       1212           JMP     CONSO
                    1213
                    1214
  F671 0604         1215 PRINT:    MVI     B,4
  F673 4E           1216 P1:       MOV     C,M
  F674 CD8BF0       1217           CALL    CONSO
  F677 23           1218           INX     H
  F678 05           1219           DCR     B
  F679 C273F6       1220           JNZ     P1
  F67C 0E20         1221           MVI     C,20H
  F67E C38BF0       1222           JMP     CONSO
                    1223
  F681 7A           1224 XTRACT:   MOV     A,D
  F682 E638         1225           ANI     38H
  F684 0F           1226           RRC
  F685 0F           1227           RRC
  F686 0F           1228           RRC
  F687 C9           1229           RET
                    1230
  F688 CD81F6       1231 CCPRNT:   CALL    XTRACT
  F68B 87           1232           ADD     A
  F68C 4F           1233           MOV     C,A
  F68D 2134F9       1234           LXI     H,CCODE
  F690 09           1235           DAD     B
  F691 4E           1236           MOV     C,M
  F692 CD8BF0       1237           CALL    CONSO
  F695 23           1238           INX     H
  F696 4E           1239           MOV     C,M
  F697 CD8BF0       1240           CALL    CONSO
  F69A 0E20         1241           MVI     C,20H
  F69C CD8BF0       1242           CALL    CONSO
  F69F C38BF0       1243           JMP     CONSO
                    1244
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

F6A2 CD81F6       1245 RPPRNT: CALL    XTRACT
F6A5 E606         1246         ANI     6
F6A7 FE06         1247         CPI     6
F6A9 C25AF6       1248         JNZ     RGPRNT
F6AC 0E53         1249         MVI     C,53H
F6AE CD8BF0       1250         CALL    CONSO
F6B1 0E50         1251         MVI     C,50H
F6B3 C38BF0       1252         JMP     CONSO
                  1253
F6B6 0D           1254 INSPT:  DCR     C
F6B7 CD04F4       1255         CALL    EXPR    ;GET STARTING ADDRESS
F6BA CD2FF6       1256 DISASM: CALL    TCRET
F6BD E1           1257         POP     H
F6BE CD44F9       1258         CALL    TCSTS   ;TEST FOR COMPLETION
F6C1 E5           1259         PUSH    H
F6C2 CD04F6       1260         CALL    THXW    ;PRINT <PC>
F6C5 0E20         1261         MVI     C,20H
F6C7 CD8BF0       1262         CALL    CONSO
F6CA CD8BF0       1263         CALL    CONSO
F6CD CD53F6       1264         CALL    RDBYTE  ;GET DATA
F6D0 57           1265         MOV     D,A
F6D1 2137F8       1266         LXI     H,TABLE ;GO AND PRINT SOURCE
F6D4 011100       1267         LXI     B,11H
F6D7 BE           1268 D1:     CMP     M
F6D8 CA29F8       1269         JZ      TG1
F6DB 23           1270         INX     H
F6DC 0D           1271         DCR     C
F6DD C2D7F6       1272         JNZ     D1
F6E0 0E0A         1273         MVI     C,0AH
F6E2 BE           1274 D2:     CMP     M
F6E3 CA15F8       1275         JZ      TG2
F6E6 23           1276         INX     H
F6E7 0D           1277         DCR     C
F6E8 C2E2F6       1278         JNZ     D2
F6EB 0E06         1279         MVI     C,6
F6ED BE           1280 D3:     CMP     M
F6EE CAF9F7       1281         JZ      TG3
F6F1 23           1282         INX     H
F6F2 0D           1283         DCR     C
F6F3 C2EDF6       1284         JNZ     D3
F6F6 E6C0         1285         ANI     0C0H
F6F8 FE40         1286         CPI     40H
F6FA CADFF7       1287         JZ      MG0
F6FD FE80         1288         CPI     80H
F6FF CAD0F7       1289         JZ      MG1
F702 7A           1290         MOV     A,D
F703 E6C7         1291         ANI     0C7H
F705 D604         1292         SUI     4
F707 CAC1F7       1293         JZ      MG2
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

F70A 3D           1294          DCR     A
F70B CA8BF7       1295          JZ      MG3
F70E 3D           1296          DCR     A
F70F CAA7F7       1297          JZ      MG4
F712 7A           1298          MOV     A,D
F713 E6C0         1299          ANI     0C0H
F715 CA7BF7       1300          JZ      MG5
F718 7A           1301          MOV     A,D
F719 E6C7         1302          ANI     0C7H
F71B D6C0         1303          SUI     0C0H
F71D CA70F7       1304          JZ      MG6
F720 D602         1305          SUI     2
F722 CA65F7       1306          JZ      MG7
F725 D602         1307          SUI     2
F727 CA5AF7       1308          JZ      MG8
F72A D603         1309          SUI     3
F72C CA4BF7       1310          JZ      MG9
F72F 7A           1311          MOV     A,D
F730 E607         1312          ANI     07
F732 4F           1313          MOV     C,A
F733 212BF9       1314          LXI     H,PPOP-1
F736 09           1315          DAD     B
F737 CD71F6       1316          CALL    PRINT
F73A CD81F6       1317          CALL    XTRACT
F73D FE06         1318          CPI     6
F73F C2CAF7       1319          JNZ     D6
F742 2128F9       1320          LXI     H,PPSW
F745 CD71F6       1321          CALL    PRINT
F748 C3BAF6       1322          JMP     DISASM
                  1323
F74B 2124F9       1324 MG9:     LXI     H,PRST
F74E CD71F6       1325          CALL    PRINT
F751 CD81F6       1326          CALL    XTRACT
F754 CD9BF5       1327          CALL    THXB
F757 C3BAF6       1328          JMP     DISASM
                  1329
F75A 0E43         1330 MG8:     MVI     C,43H
F75C CD8BF0       1331          CALL    CONSO
F75F CD88F6       1332          CALL    CCPRNT
F762 C304F8       1333          JMP     D7
F765 0E4A         1334 MG7:     MVI     C,4AH
F767 CD8BF0       1335          CALL    CONSO
F76A CD88F6       1336          CALL    CCPRNT
F76D C304F8       1337          JMP     D7
F770 0E52         1338 MG6:     MVI     C,52H
F772 CD8BF0       1339          CALL    CONSO
F775 CD88F6       1340          CALL    CCPRNT
F778 C3BAF6       1341          JMP     DISASM
                  1342
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

F77B 210CF9       1343 MG5:    LXI     H,PLXI
F77E 7A           1344         MOV     A,D
F77F E60F         1345         ANI     0FH
F781 3D           1346         DCR     A
F782 CA99F7       1347         JZ      MG51
F785 FE04         1348         CPI     4
F787 DA8CF7       1349         JC      D4
F78A D605         1350         SUI     5
F78C 87           1351 D4:     ADD     A
F78D 87           1352         ADD     A
F78E 4F           1353         MOV     C,A
F78F 09           1354         DAD     B
F790 CD71F6       1355         CALL    PRINT
F793 CDA2F6       1356         CALL    RPPRNT
F796 C3BAF6       1357         JMP     DISASM
                  1358
F799 CD71F6       1359 MG51:   CALL    PRINT
F79C CDA2F6       1360         CALL    RPPRNT
F79F 0E2C         1361         MVI     C,2CH
F7A1 CD8BF0       1362         CALL    CONSO
F7A4 C304F8       1363         JMP     D7
F7A7 2108F9       1364 MG4:    LXI     H,PMVI
F7AA CD71F6       1365         CALL    PRINT
F7AD CD81F6       1366         CALL    XTRACT
F7B0 CD5AF6       1367         CALL    RGPRNT
F7B3 0E2C         1368         MVI     C,2CH
F7B5 CD8BF0       1369         CALL    CONSO
F7B8 C320F8       1370         JMP     D8
                  1371
F7BB 2104F9       1372 MG3:    LXI     H,PDCR
F7BE C3C4F7       1373         JMP     D5
F7C1 2100F9       1374 MG2:    LXI     H,PINR
F7C4 CD71F6       1375 D5:     CALL    PRINT
F7C7 CD81F6       1376         CALL    XTRACT
F7CA CD5AF6       1377 D6:     CALL    RGPRNT
F7CD C3BAF6       1378         JMP     DISASM
                  1379
F7D0 7A           1380 MG1:    MOV     A,D
F7D1 E638         1381         ANI     38H
F7D3 0F           1382         RRC
F7D4 4F           1383         MOV     C,A
F7D5 21E0F8       1384         LXI     H,PADD
F7D8 09           1385         DAD     B
F7D9 CD71F6       1386         CALL    PRINT
F7DC C3F0F7       1387         JMP     D9
F7DF 21DCF8       1388 MG0:    LXI     H,PMOV
F7E2 CD71F6       1389         CALL    PRINT
F7E5 CD81F6       1390         CALL    XTRACT
F7E8 CD5AF6       1391         CALL    RGPRNT
```

```
  LOC  OBJ          SEQ          SOURCE STATEMENT

  F7EB 0E2C         1392          MVI     C,2CH
  F7ED CD8BF0       1393          CALL    CONSO
  F7F0 7A           1394 D9:      MOV     A,D
  F7F1 E607         1395          ANI     7
  F7F3 CD5AF6       1396          CALL    RGPRNT
  F7F6 C3BAF6       1397          JMP     DISASM
                    1398
  F7F9 79           1399 TG3:     MOV     A,C
  F7FA 87           1400          ADD     A
  F7FB 87           1401          ADD     A
  F7FC 4F           1402          MOV     C,A
  F7FD 21C0F8       1403          LXI     H,TAB3-4
  F800 09           1404          DAD     B
  F801 CD71F6       1405          CALL    PRINT
  F804 CD53F6       1406 D7:      CALL    RDBYTE
  F807 57           1407          MOV     D,A
  F808 CD53F6       1408          CALL    RDBYTE
  F80B CD9BF5       1409          CALL    THXB
  F80E 7A           1410          MOV     A,D
  F80F CD9BF5       1411          CALL    THXB
  F812 C3BAF6       1412          JMP     DISASM
                    1413
  F815 79           1414 TG2:     MOV     A,C
  F816 87           1415          ADD     A
  F817 87           1416          ADD     A
  F818 4F           1417          MOV     C,A
  F819 2198F8       1418          LXI     H,TAB2-4
  F81C 09           1419          DAD     B
  F81D CD71F6       1420          CALL    PRINT
  F820 CD53F6       1421 D8:      CALL    RDBYTE
  F823 CD9BF5       1422          CALL    THXB
  F826 C3BAF6       1423          JMP     DISASM
                    1424
  F829 79           1425 TG1:     MOV     A,C
  F82A 87           1426          ADD     A
  F82B 87           1427          ADD     A
  F82C 4F           1428          MOV     C,A
  F82D 2154F8       1429          LXI     H,TAB1-4
  F830 09           1430          DAD     B
  F831 CD71F6       1431          CALL    PRINT
  F834 C3BAF6       1432          JMP     DISASM
                    1433
  F837 00           1434 TABLE:   DB      0,7,0FH,17H
  F838 07
  F839 0F
  F83A 17
  F83B 1F           1435          DB      1FH,27H,2FH,37H
  F83C 27
  F83D 2F
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

F83E 37
F83F 3F           1436         DB      3FH,76H,0C9H,0E3H
F840 76
F841 C9
F842 E3
F843 E9           1437         DB      0E9H,0EBH,0F3H,0F9H
F844 EB
F845 F3
F846 F9
F847 FB           1438         DB      0FBH,0C6H,0CEH,0D3H
F848 C6
F849 CE
F84A D3
F84B D6           1439         DB      0D6H,0DBH,0DEH,0E6H
F84C DB
F84D DE
F84E E6
F84F EE           1440         DB      0EEH,0F6H,0FEH,22H
F850 F6
F851 FE
F852 22
F853 2A           1441         DB      2AH,32H,3AH,0C3H,0CDH
F854 32
F855 3A
F856 C3
F857 CD
F858 45492020     1442 TAB1:   DB      'EI  ','SPHL','DI  ','XCHG'
F85C 5350484C
F860 44492020
F864 58434847
F868 5043484C     1443         DB      'PCHL','XTHL','RET ','HLT '
F86C 5854484C
F870 52455420
F874 484C5420
F878 434D4320     1444         DB      'CMC ','STC ','CMA ','DAA '
F87C 53544320
F880 434D4120
F884 44414120
F888 52415220     1445         DB      'RAR ','RAL ','RRC ','RLC '
F88C 52414C20
F890 52524320
F894 524C4320
F898 4E4F5020     1446         DB      'NOP '
F89C 43504920     1447 TAB2:   DB      'CPI ','ORI ','XRI ','ANI '
F8A0 4F524920
F8A4 58524920
F8A8 414E4920
F8AC 53424920     1448         DB      'SBI ','IN  ','SUI ','OUT '
F8B0 494E2020
```

```
 LOC  OBJ          SEQ          SOURCE STATEMENT

 F8B4 53554920
 F8B8 4F555420
 F8BC 41434920     1449          DB      'ACI ','ADI '
 F8C0 41444920
 F8C4 43414C4C     1450 TAB3:    DB      'CALL','JMP ','LDA ','STA '
 F8C8 4A4D5020
 F8CC 4C444120
 F8D0 53544120
 F8D4 4C484C44     1451          DB      'LHLD','SHLD'
 F8D8 53484C44
 F8DC 4D4F5620     1452 PMOV:    DB      'MOV '
 F8E0 41444420     1453 PADD:    DB      'ADD ','ADC ','SUB ','SBB '
 F8E4 41444320
 F8E8 53554220
 F8EC 53424220
 F8F0 414E4120     1454          DB      'ANA ','XRA ','ORA ','CMP '
 F8F4 58524120
 F8F8 4F524120
 F8FC 434D5020
 F900 494E5220     1455 PINR:    DB      'INR '
 F904 44435220     1456 PDCR:    DB      'DCR '
 F908 4D564920     1457 PMVI:    DB      'MVI '
 F90C 4C584920     1458 PLXI:    DB      'LXI ','STAX','INX ','DAD '
 F910 53544158
 F914 494E5820
 F918 44414420
 F91C 4C444158     1459          DB      'LDAX','DCX '
 F920 44435820
 F924 52535420     1460 PRST:    DB      'RST '
 F928 50535720     1461 PPSW:    DB      'PSW '
 F92C 504F5020     1462 PPOP:    DB      'POP ','PUSH'
 F930 50555348
 F934 4E5A         1463 CCODE:   DB      'NZ','Z ','NC','C '
 F936 5A20
 F938 4E43
 F93A 4320
 F93C 504F         1464          DB      'PO','PE','P ','M '
 F93E 5045
 F940 5020
 F942 4D20
                   1465
                   1466
                   1467
                   1468
                   1469
                   1470 TCSTS:   ;SUBROUTINE TO TEST FOR KEYBOARD INTRPT
 F944 CDF4F0       1471          CALL    CSTS
 F947 FE00         1472          CPI     0
 F949 C8           1473          RZ                    ;NO, KEEP GOING
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

F94A 33           1474          INX     SP
F94B 33           1475          INX     SP
F94C C39CF1       1476          JMP     START
                  1477
                  1478 PTAPE:   ;SUBROUTINE TO COPY PAPER TAPES ETC
F94F CDDFF0       1479          CALL    RDR      ;GET CHARACTER
F952 4F           1480          MOV     C,A      ;SET UP TO
F953 CDA0F0       1481          CALL    PUNCH    ;OUTPUT CHARACTER
F956 CD44F9       1482          CALL    TCSTS    ;GET CONSOLE STATUS
F959 C34FF9       1483          JMP     PTAPE    ;NEXT
                  1484
                  1485
                  1486 TIMER:   ;SUB TO DELAY
                  1487                   ;<D/E> = DELAY TIME
F95C 1D           1488          DCR     E
F95D C25CF9       1489          JNZ     TIMER
F960 15           1490          DCR     D
F961 C25CF9       1491          JNZ     TIMER
F964 C9           1492          RET
                  1493
                  1494
                  1495 ;    *** L P R C ***
                  1496 DUFG:    ;ROUTINE TO DISPLAY I/O CONFIGURATION
                  1497
                  1498 GET      MACRO   DATA
                  1499          LXI     H,DATA
                  1500          CALL    MSG
                  1501          LDA     IOCW
                  1502          ENDM
                  1503
F965 CD2FF6       1504          CALL    TCRET
                  1505          GET     UM1
F968 21A7F9       1506+         LXI     H,UM1
F96B CD48F6       1507+         CALL    MSG
F96E 3A0300       1508+         LDA     IOCW
F971 CDC3F9       1509          CALL    DUFG1
                  1510          GET     UM2
F974 21ABF9       1511+         LXI     H,UM2
F977 CD48F6       1512+         CALL    MSG
F97A 3A0300       1513+         LDA     IOCW
F97D CDBDF9       1514          CALL    DUFG2
                  1515          GET     UM3
F980 21AFF9       1516+         LXI     H,UM3
F983 CD48F6       1517+         CALL    MSG
F986 3A0300       1518+         LDA     IOCW
F989 CDBBF9       1519          CALL    DUFG3
                  1520          GET     UM4
F98C 21B3F9       1521+         LXI     H,UM4
F98F CD48F6       1522+         CALL    MSG
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

F992 3A0300       1523+        LDA     IOCW
F995 CDC1F9       1524         CALL    DUFG4
F998 21B7F9       1525         LXI     H,UM5
F99B CD48F6       1526         CALL    MSG
F99E 3A0B00       1527         LDA     VDMCW
F9A1 CDC1F9       1528         CALL    DUFG4
F9A4 C39CF1       1529         JMP     START
                  1530
                  1531
F9A7 20433D       1532 UM1:    DB      ' C=',-1
F9AA FF
F9AB 20523D       1533 UM2:    DB      ' R=',-1
F9AE FF
F9AF 20503D       1534 UM3:    DB      ' P=',-1
F9B2 FF
F9B3 204C3D       1535 UM4:    DB      ' L=',-1
F9B6 FF
F9B7 20563D       1536 UM5:    DB      ' V=',-1
F9BA FF
                  1537
F9BB 0F           1538 DUFG3:  RRC             ;SHIFT R 4 PLACES
F9BC 0F           1539         RRC
F9BD 0F           1540 DUFG2:  RRC             ;SHIFT R 2 PLACES
F9BE 0F           1541         RRC
F9BF 0F           1542         RRC             ;NOW COMPENSATE FOR RLC'S
F9C0 0F           1543         RRC
F9C1 07           1544 DUFG4:  RLC             ;SHIFT L 2 PLACES
F9C2 07           1545         RLC
F9C3 E603         1546 DUFG1:  ANI     3       ;MASK
F9C5 C39BF5       1547         JMP     THXB    ;AND PRINT IT
                  1548
                  1549
                  1550 CNFG:   ;SUB TO CONFIGURE I/O STATE
F9C8 CD2DF1       1551         CALL    CHIN
F9CB F5           1552         PUSH    PSW
F9CC 0E20         1553         MVI     C,' '
F9CE CD8BF0       1554         CALL    CONSO
F9D1 F1           1555         POP     PSW
F9D2 FE56         1556         CPI     'V'
F9D4 CA2FFA       1557         JZ      CVID
F9D7 FE4C         1558         CPI     'L'
F9D9 CA1EFA       1559         JZ      CLO
F9DC FE50         1560         CPI     'P'
F9DE CA0BFA       1561         JZ      CPUN
F9E1 FE52         1562         CPI     'R'
F9E3 CAFAF9       1563         JZ      CRI
F9E6 FE43         1564         CPI     'C'
F9E8 C2BEF1       1565         JNZ     ERROR
F9EB 3A0300       1566         LDA     IOCW
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

F9EE CDC3F9       1567          CALL    DUFG1   ;PRINT CURRENT STATE
F9F1 CD48FA       1568          CALL    GETN
F9F4 47           1569          MOV     B,A
F9F5 3EFC         1570          MVI     A,11111100B
F9F7 C35FFA       1571          JMP     CNFG6
F9FA 3A0300       1572 CRI:     LDA     IOCW
F9FD CDBDF9       1573          CALL    DUFG2
FA00 CD48FA       1574          CALL    GETN    ;GET NEW VALUE
FA03 07           1575          RLC
FA04 07           1576          RLC             ;MOVE INTO POSITION
FA05 47           1577          MOV     B,A
FA06 3EF3         1578          MVI     A,11110011B
FA08 C35FFA       1579          JMP     CNFG6
FA0B 3A0300       1580 CPUN:    LDA     IOCW
FA0E CDBBF9       1581          CALL    DUFG3
FA11 CD48FA       1582          CALL    GETN
FA14 07           1583          RLC
FA15 07           1584          RLC
FA16 07           1585          RLC
FA17 07           1586          RLC
FA18 47           1587          MOV     B,A
FA19 3ECF         1588          MVI     A,11001111B
FA1B C35FFA       1589          JMP     CNFG6
FA1E 3A0300       1590 CLO:     LDA     IOCW
FA21 CDC1F9       1591          CALL    DUFG4
FA24 CD48FA       1592          CALL    GETN
FA27 0F           1593          RRC
FA28 0F           1594          RRC
FA29 47           1595          MOV     B,A
FA2A 3E3F         1596          MVI     A,00111111B
FA2C C35FFA       1597          JMP     CNFG6
FA2F 3A0B00       1598 CVID:    LDA     VDMCW
FA32 07           1599          RLC
FA33 07           1600          RLC
FA34 CDC3F9       1601          CALL    DUFG1
FA37 CD48FA       1602          CALL    GETN
FA3A 0F           1603          RRC
FA3B 0F           1604          RRC
FA3C 47           1605          MOV     B,A
FA3D 210B00       1606          LXI     H,VDMCW
FA40 7E           1607          MOV     A,M
FA41 E63F         1608          ANI     00111111B
FA43 B0           1609          ORA     B
FA44 77           1610          MOV     M,A
FA45 C39CF1       1611          JMP     START
                  1612
FA48 215CFA       1613 GETN:    LXI     H,UM6
FA4B CD48F6       1614          CALL    MSG
FA4E CD2DF1       1615          CALL    CHIN
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

FA51 D630         1616          SUI     30H
FA53 FABEF1       1617          JM      ERROR
FA56 FE04         1618          CPI     4
FA58 F2BEF1       1619          JP      ERROR
FA5B C9           1620          RET
FA5C 2D20         1621 UM6:     DB      '- ',-1
FA5E FF
                  1622
FA5F 210300       1623 CNFG6:   LXI     H,IOCW
FA62 A6           1624          ANA     M         ;MASK IT
FA63 B0           1625          ORA     B         ;UPDATE
FA64 77           1626          MOV     M,A       ;AND SAVE
FA65 C39CF1       1627          JMP     START
                  1628
                  1629
                  1630 PINP:    ;SUB TO INPUT AND DISPLAY <PORT>
FA68 0D           1631          DCR     C
FA69 CD04F4       1632          CALL    EXPR      ;GET PORT NUMBER
FA6C 215CFA       1633          LXI     H,UM6     ;LOCATE "-" SIGN
FA6F CD48F6       1634          CALL    MSG       ;OUTPUT IT
FA72 C1           1635          POP     B         ;
FA73 210500       1636          LXI     H,IOEX
FA76 36DB         1637          MVI     M,0DBH    ;SET UP INPUT INSTR
FA78 23           1638          INX     H
FA79 71           1639          MOV     M,C       ;PUT IN PORT NUMBER
FA7A 23           1640          INX     H
FA7B 36C9         1641          MVI     M,0C9H    ;SET UP RETURN
FA7D CD0500       1642          CALL    IOEX      ;DO IT
FA80 CD9BF5       1643          CALL    THXB      ;OUTPUT IT
FA83 C39CF1       1644          JMP     START
                  1645
                  1646
                  1647 POUT:    ;SUB TO OUTPUT TO PORT
FA86 CD04F4       1648          CALL    EXPR      ;GET INSRTUCTIONS
FA89 C1           1649          POP     B         ;POP DATA
FA8A D1           1650          POP     D         ;POP PORT NO
FA8B 210500       1651          LXI     H,IOEX
FA8E 36D3         1652          MVI     M,0D3H    ;SET UP INSTR
FA90 23           1653          INX     H
FA91 73           1654          MOV     M,E
FA92 23           1655          INX     H
FA93 36C9         1656          MVI     M,0C9H
FA95 79           1657          MOV     A,C       ;SET UP TO
FA96 CD0500       1658          CALL    IOEX
FA99 C39CF1       1659          JMP     START
                  1660
                  1661 VERY:    ;ROUTINE TO VERIFY BLOCK OF MEMORY
                  1662          ;DOES NOT DESTROY ANY DATA
FA9C CD04F4       1663          CALL    EXPR      ;GET LIMITS
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

FA9F D1           1664           POP     D        ;GET HI
FAA0 E1           1665           POP     H        ;GET LOW
FAA1 CD2FF6       1666           CALL    TCRET
FAA4 CD44F9       1667 VERY1:    CALL    TCSTS    ;TEST  FOR INTRPT
FAA7 46           1668           MOV     B,M      ;SAVE <MEMORY>
FAA8 3E55         1669           MVI     A,55H    ;SET UP BIT PATTERN
FAAA 77           1670           MOV     M,A      ;PUT IN MEMORY
FAAB BE           1671           CMP     M        ;TEST FOR WRITE
FAAC C2BFFA       1672           JNZ     VERY2
FAAF 2F           1673           CMA
FAB0 77           1674           MOV     M,A      ;INVERT IT
FAB1 BE           1675           CMP     M        ;AND TEST
FAB2 C2BFFA       1676           JNZ     VERY2
FAB5 70           1677 VERY3:    MOV     M,B      ;REPLACE ORIG DATA
FAB6 CD38F4       1678           CALL    DONE     ;TEST FOR COMPLETION
FAB9 DA9CF1       1679           JC      START
FABC C3A4FA       1680           JMP     VERY1    ;GOTO IT
FABF CD04F6       1681 VERY2:    CALL    THXW     ;PRINT BAD LOCATION
FAC2 CD2FF6       1682           CALL    TCRET
FAC5 C3B5FA       1683           JMP     VERY3    ;KEEP GOING
                  1684
                  1685 BACON:    ;SUBROUTINE FOR BAUDOT TO ASCII CONVERSION
                  1686           ;  RETURNS WITH <A> = CHARACTER
                  1687           ;  LETTERS/FIGS CHAR SETS CARRY BIT
FAC8 C5           1688           PUSH    B
FAC9 D5           1689           PUSH    D
FACA E5           1690           PUSH    H        ;SAVE <REGISTERS>
FACB FE1B         1691           CPI     33Q      ;TEST FOR "FIGS"
FACD CAEDFA       1692           JZ      FIGS
FAD0 FE1F         1693           CPI     37Q      ;TEST FOR "LETS"
FAD2 CAF2FA       1694           JZ      LETS
FAD5 110300       1695           LXI     D,CASE
FAD8 CD48FB       1696           CALL    LOCMB    ;LOCATE CASE MEMORY
FADB 4F           1697           MOV     C,A      ;SAVE
FADC 3E20         1698           MVI     A,20H    ;SET UP MASK
FADE A6           1699           ANA     M        ;MASK CASE BIT
FADF B1           1700           ORA     C        ;NOW SET TABLE INDEX
FAE0 010000       1701           LXI     B,0      ;CLEAR REGISTERS
FAE3 4F           1702           MOV     C,A      ;SET UP FOR ADD
FAE4 2104FB       1703           LXI     H,BCN1   ;POINT TO BASE TABLE
FAE7 09           1704           DAD     B        ;POINT TO CHARACTER
FAE8 7E           1705           MOV     A,M      ;AND FETCH IT
FAE9 B7           1706           ORA     A        ;CLEAR CARRY
FAEA C300FB       1707           JMP     BAC2
FAED 0E20         1708 FIGS:     MVI     C,20H    ;SET CASE MEMORY
FAEF C3F4FA       1709           JMP     $+5
FAF2 0E00         1710 LETS:     MVI     C,0      ;RESET CASE MEMORY
FAF4 110300       1711           LXI     D,CASE
FAF7 CD48FB       1712           CALL    LOCMB    ;LOCATE CASE MEMORY
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

FAFA 3EDF         1713          MVI     A,0DFH  ;SET UP TO
FAFC A6           1714          ANA     M       ;MASK CASE BIT
FAFD B1           1715          ORA     C       ;NOW SET IT UP
FAFE 77           1716          MOV     M,A     ;AND SAVE
FAFF 37           1717          STC
FB00 E1           1718 BAC2:    POP     H
FB01 D1           1719          POP     D
FB02 C1           1720          POP     B
FB03 C9           1721          RET
                  1722
                  1723          ;LOOK UP TABLE FOR CODE CONVERSION
FB04 00           1724 BCN1:    DB      0,'E',0AH,'A'
FB05 45
FB06 0A
FB07 41
FB08 20534955     1725          DB      ' SIU'
FB0C 0D           1726          DB      0DH     ;CR
FB0D 44524A4E     1727          DB      'DRJNF'
FB11 46
FB12 434B545A     1728          DB      'CKTZL'
FB16 4C
FB17 57485950     1729          DB      'WHYPQ'
FB1B 51
FB1C 4F4247       1730          DB      'OBG'
FB1F 00           1731          DB      0       ;FIGS
FB20 4D5856       1732          DB      'MXV'
FB23 00           1733          DB      0       ;LETS
FB24 00           1734          DB      0       ;BLANK
FB25 33           1735          DB      '3'
FB26 0A           1736          DB      0AH     ;LF
FB27 2D20         1737          DB      '- ',7  ;7= BELL
FB29 07
FB2A 3837         1738          DB      '87'
FB2C 0D           1739          DB      0DH     ;CR
FB2D 2434         1740          DB      '$4',27H,',!:'
FB2F 27
FB30 2C213A
FB33 28352229     1741          DB      '(5")2'
FB37 32
FB38 1B           1742          DB      1BH     ;ESC KEY = ^H
FB39 36303139     1743          DB      '6019?'
FB3D 3F
FB3E 26           1744          DB      '&'
FB3F FF           1745          DB      0FFH    ;FIGS
FB40 2E2F3B       1746          DB      './;'
FB43 FF           1747 BCN2:    DB      0FFH    ;LETS
                  1748
                  1749 IRST:    ;SUB TO RESET INTERRUPTS
FB44 D3BA         1750          OUT     ICP     ;RESET 8214 CHIP
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

FB46 FB           1751          EI
FB47 C9           1752          RET
                  1753
                  1754 LOCMB:   ;SUB TO LOCATE CONTROL  BYTE IN RAM BLOCK
                  1755                                ;<D/E> = INDEX
FB48 2A0E00       1756          LHLD    RBADR   ;LOCATE BASE OF RAM BLOCK
FB4B 19           1757          DAD     D       ;POINT TO IT
FB4C C9           1758          RET
                  1759
                  1760 ASCBD:   ;ROUTINE TO CONVERT ASCII TO BAUDOT
                  1761          ; <C> =  CHARACTER TO BE CONVERTED
                  1762          ;RETURNS <A> = BAUDOT CHARACTER
                  1763          ;       CARRY IS SET FOR ILLEGAL CHARACTER
                  1764          ;       BIT7 IS SET IF LETTERS SHIFT REQD
                  1765          ;       BIT6 IS SET IF NUMBERS SHIFT REQD
0080              1766 UCASE    EQU     80H     ;BIT7
0040              1767 LCASE    EQU     40H     ;BIT6
FB4D C5           1768          PUSH    B
FB4E D5           1769          PUSH    D
FB4F E5           1770          PUSH    H
FB50 79           1771          MOV     A,C     ;FETCH IT
                  1772                          ;TEST FOR CR LF SP
                  1773                          ;IF SUCCESSFUL, BYPASS
                  1774                          ;CASE/TABLE
FB51 FE20         1775          CPI     20H
FB53 CA65FB       1776          JZ      ASC10   ;SP
FB56 FE0A         1777          CPI     0AH
FB58 CA6AFB       1778          JZ      ASCB9   ;LF
FB5B FE0D         1779          CPI     0DH
FB5D C26FFB       1780          JNZ     ASCB0   ;FAILED, GO TO TABLE
FB60 3E08         1781          MVI     A,8     ;CR
FB62 C3B3FB       1782          JMP     ASCB8
FB65 3E04         1783 ASC10:   MVI     A,4
FB67 C3B3FB       1784          JMP     ASCB8
FB6A 3E02         1785 ASCB9:   MVI     A,2
FB6C C3B3FB       1786          JMP     ASCB8
FB6F FE60         1787 ASCB0:   CPI     60H     ;TEST FOR LOWER CASE
FB71 DA76FB       1788          JC      ASCB1   ;NOPE IGNORE
FB74 D620         1789          SUI     20H     ;CONVERT TO UPPER CASE
FB76 4F           1790 ASCB1:   MOV     C,A     ;SAVE IT
FB77 063F         1791          MVI     B,BCN2-BCN1
FB79 2143FB       1792          LXI     H,BCN2 ;POINT TO TOP OF TABLE
FB7C 7E           1793 ASCB2:   MOV     A,M     ;PREPARE TO
FB7D B9           1794          CMP     C       ;SEARCH THE TABLE
FB7E CA8DFB       1795          JZ      ASCB4   ;MATCH
FB81 05           1796          DCR     B
FB82 CA89FB       1797          JZ      ASCB3   ;OOPS OUT OF TABLE
FB85 2B           1798          DCX     H
FB86 C37CFB       1799          JMP     ASCB2   ;KEEP TRYING
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

                  1800
FB89 37           1801 ASCB3:  STC             ;NON VALID CHARACTER
FB8A C3B3FB       1802         JMP     ASCB8
                  1803
FB8D FE40         1804 ASCB4:  CPI     40H     ;TEST ASCII CHAR FOR
                  1805                         ;       LETTERS CASE
FB8F D297FB       1806         JNC     ASCB5   ;UPPER
FB92 0E40         1807         MVI     C,LCASE
FB94 C399FB       1808         JMP     ASCB6
FB97 0E80         1809 ASCB5:  MVI     C,UCASE
FB99 110300       1810 ASCB6:  LXI     D,CASE  ;DETERMINE OF CASE = SAME
FB9C CD48FB       1811         CALL    LOCMB
FB9F 3EC0         1812         MVI     A,0C0H  ;MASK CASE BITS
FBA1 A6           1813         ANA     M       ;SET TO OLD CASE
FBA2 57           1814         MOV     D,A     ;SAVE OLD CASE
FBA3 7E           1815         MOV     A,M     ;GET OLD BYTE
FBA4 E63F         1816         ANI     3FH     ;MASK OUT CASE BITS
FBA6 B1           1817         ORA     C       ;REPLACE BY NEW
FBA7 77           1818         MOV     M,A     ;AND SAVE
FBA8 7A           1819         MOV     A,D     ;NOW GET NEW CASE
FBA9 B9           1820         CMP     C       ;TEST FOR EQUALITY
FBAA C2AFFB       1821         JNZ     ASCB7   ;THEY ARE DIFFERENT
FBAD 0E00         1822         MVI     C,0     ;RESET FLAG BITS
FBAF 78           1823 ASCB7:  MOV     A,B     ;GET BAUDOT EQUIV
FBB0 E61F         1824         ANI     1FH     ;MASK OUT 5 BITS
FBB2 B1           1825         ORA     C       ;SET CASE FLAG BITS
FBB3 E1           1826 ASCB8:  POP     H
FBB4 D1           1827         POP     D
FBB5 C1           1828         POP     B
FBB6 C9           1829         RET
                  1830
                  1831 JAMT:   ;ROUTINE TO PUT ASCII TEXT INTO MEMORY
                  1832                         ;FROM CONSOLE
FBB7 0D           1833         DCR     C
FBB8 CD04F4       1834         CALL    EXPR    ;GET MEMORY LOCATION
FBBB E1           1835         POP     H
FBBC CD2FF6       1836         CALL    TCRET
FBBF 0E3D         1837         MVI     C,'='
FBC1 CD8BF0       1838         CALL    CONSO
FBC4 CD2DF1       1839 JAMT2:  CALL    CHIN    ;FETCH CHARACTER
FBC7 FE0D         1840         CPI     0DH     ;IS IT CARRIAGE RETURN
FBC9 CAD6FB       1841         JZ      JAMT1   ;YES, TERMINATE SEQUENCE
FBCC FE08         1842         CPI     08H     ;IS IT BACKSPACE
FBCE CADBFB       1843         JZ      JAMT3   ;YES BACK UP TEXT
FBD1 77           1844         MOV     M,A     ;SAVE CHARACTER
FBD2 23           1845         INX     H
FBD3 C3C4FB       1846         JMP     JAMT2   ;GO GET NEXT ONE
                  1847
FBD6 36FF         1848 JAMT1:  MVI     M,-1    ;SET UP END OF TEXT CHARAC
```

```
  LOC  OBJ          SEQ          SOURCE STATEMENT

  FBD8 C39CF1       1849          JMP     START   ;RETURN TO AMS80
                    1850
  FBDB 2B           1851 JAMT3:   DCX     H       ;BACK UP POINTER
  FBDC C3C4FB       1852          JMP     JAMT2   ;GO GET NEXT CHARACTER
                    1853
                    1854 AOUT:                    ;OUTPUT <A> TO CONSOLE
  FBDF C5           1855          PUSH    B       ;SAVE <C>
  FBE0 CD8AF0       1856          CALL    CONSA   ;OUTPUT TO CONSOLE
  FBE3 C1           1857          POP     B       ;RETRIEVE
  FBE4 C9           1858          RET
                    1859
                    1860 CHINS:                   ;INPUT AND ECHO
                    1861                          ;IGNORE ANY SPACE CHARACTER
  FBE5 CD2DF1       1862          CALL    CHIN    ;INPUT BIT
  FBE8 FE20         1863          CPI     ' '     ;IS IT SPACE
  FBEA CAE5FB       1864          JZ      CHINS   ;YES, IGNORE IT
  FBED C9           1865          RET             ;NO, PASS IT
                    1866
                    1867 ATTSGN:                  ;RECOVER INTS &
  FBEE F5           1868          PUSH    PSW
  FBEF 3E02         1869          MVI     A,2     ;SET LEVEL
  FBF1 CD44FB       1870          CALL    IRST    ;RESET INTS
  FBF4 F1           1871          POP     PSW
  FBF5 C3E5F4       1872          JMP     GOTO    ;AND "G" IT
                    1873
  FBF8 F3           1874 DOS:     DI              ;INTS OFF FOR DISC
  FBF9 C300E8       1875          JMP     0E800H  ;NORTHSTAR DISC BOOT
                    1876                          ;OK FOR SD,DD AND CP/M
                    1877
  FBFC              1878 PROM     SET     $
                    1879          END

PUBLIC SYMBOLS


EXTERNAL SYMBOLS


USER SYMBOLS
ACTBL  A F541    AMS80  A F000    AOUT   A FBDF    ASAV   A 0005
ASC1   A F365    ASC10  A FB65    ASC2   A F348    ASC3   A F33A
ASCB0  A FB6F    ASCB1  A FB76    ASCB2  A FB7C    ASCB3  A FB89
ASCB4  A FB8D    ASCB5  A FB97    ASCB6  A FB99    ASCB7  A FBAF
ASCB8  A FBB3    ASCB9  A FB6A    ASCBD  A FB4D    ASCI   A F335
ATTSGN A FBEE    BAC2   A FB00    BACON  A FAC8    BASE   A F000
BCN1   A FB04    BCN2   A FB43    BEGIN  A F14D    BG0    A F150
BG1    A F160    BOUT   + 0001    BSAV   A 0003    BTTI   A FC48
BTTO   A FC4B    BYTST  A 0000    CASE   A 0003    CCODE  A F934
CCURNT A F688    CHIN   A F12D    CHINS  A FBE5    CLO    A FA1E
```

CNFG	A	F9C8	CNFG6	A	FA5F	CONI0	A	FC39	CONI1	A	FC36
CONI2	A	FC33	CONI3	A	FC30	CONO0	A	FC2D	CONO1	A	FC2A
CONO2	A	FC27	CONO3	A	FC24	CONSA	A	F08A	CONSB	A	F109
CONSI	A	F0CA	CONSO	A	F08B	CONV	A	F5C7	CPUN	A	FA0B
CRI	A	F9FA	CSAV	A	0002	CSTS	A	F0F4	CSTS0	A	FC45
CSTS1	A	FC42	CSTS2	A	FC3F	CSTS3	A	FC3C	CUSTOM	A	FC8B
CVID	A	FA2F	D1	A	F6D7	D2	A	F6E2	D3	A	F6ED
D4	A	F78C	D5	A	F7C4	D6	A	F7CA	D7	A	F804
D8	A	F820	D9	A	F7F0	DEBUG	A	0000	DI0	A	F22E
DIL	A	F234	DISASM	A	F6BA	DISP	A	F229	DONE	A	F438
DUS	A	FBF8	DSAV	A	0001	DUFG	A	F965	DUFG1	A	F9C3
DUFG2	A	F9BD	DUFG3	A	F9BB	DUFG4	A	F9C1	ENDX	A	F541
ERF	A	F24F	ERROR	A	F1BE	ESAV	A	0000	EX0	A	F407
EX1	A	F40A	EX2	A	F41A	EX3	A	F428	EXF	A	F430
EXIT	A	F52F	EXPR	A	F404	FALSE	A	0000	FI0	A	F21F
FINS	A	FAED	FILL	A	F218	FSAV	A	0004	GET	+	0002
GETN	A	FA48	G00	A	F4FF	G01	A	F505	G02	A	F51D
GOD	A	F527	GOTO	A	F4E5	HEAD	A	F563	HEAD1	A	F56B
HEO2	A	F57A	HELLO	A	F139	HEXM	A	F1D1	HLX	A	F536
ICMV	A	0010	ICP	A	00BA	INSPT	A	F6B6	IOCHK	A	F124
IOLW	A	0003	IOEX	A	0005	IOROM	A	FC00	IORW	A	FC50
IOSET	A	F128	IOST	+	0000	IRST	A	FB44	JAMT	A	FBB7
JAMT1	A	FBD6	JAMT2	A	FBC4	JAMT3	A	FBDB	LCASE	A	0040
LIMET	A	F625	LE0	A	F5E8	LEAD	A	F5E6	LETS	A	FAF2
LHB	A	F5B1	LHXN	A	F5BF	LHXW	A	F60C	LIST	A	F0B5
LIST0	A	FC21	LIST1	A	FC1E	LIST2	A	FC1B	LIST3	A	FC18
LOC	A	F11B	LOCMB	A	FB48	LSAV	A	000F	MEM0	A	F466
MCK	A	F452	MEMSAZ	A	F45B	MEMSIZ	A	F462	MG0	A	F7DF
MG1	A	F7D0	MG2	A	F7C1	MG3	A	F7BB	MG4	A	F7A7
MG5	A	F77B	MG51	A	F799	MG6	A	F770	MG7	A	F765
MG8	A	F75A	MG9	A	F74B	MOVE	A	F205	MSG	A	F648
NI9	A	F20C	NI0	A	F600	NIBBLE	A	F5F2	NULL	A	F26E
NXL	A	F673	P2C	A	F445	PADD	A	F8E0	PCHK	A	F442
PCRET	A	F63E	PCX	A	F539	PDCR	A	F904	PHXB	A	F580
PHXW	A	F614	PINP	A	FA68	PINR	A	F900	PLXI	A	F90C
PMOV	A	F8DC	PMVI	A	F908	POB	A	F112	POUT	A	FA86
PPOP	A	F92C	PPSW	A	F928	PRINT	A	F671	PROM	A	FBFC
PRST	A	F924	PSAV	A	0013	PTAPE	A	F94F	PUN0	A	FC09
PUN1	A	FC06	PUN2	A	FC03	PUN3	A	FC00	PUNCH	A	F0A0
RDADR	A	000E	RDBYTE	A	F653	RDR	A	F0DF	RDR0	A	FC15
RDR1	A	FC12	RDR2	A	FC0F	RDR3	A	FC0C	READ	A	F274
RED0	A	F278	RED1	A	F297	RED2	A	F2A9	RED3	A	F2B6
RESTAR	A	F477	RGP1	A	F664	RGP2	A	F66B	RGPRNT	A	F65A
RHXB	A	F5CF	RIX	A	F61C	RPPRNT	A	F6A2	RS1	A	0008
RST0	A	F486	RST1	A	F4A1	RST2	A	F4AE	RST3	A	F4AF
RST4	A	F4CE	RST5	A	F4DD	SDEHL	A	F1FE	SSAV	A	0007
STAR1	A	F19F	START	A	F19C	STPR	A	0000	SU0	A	F2C5
SU1	A	F2E4	SUBS	A	F2BA	T1A	A	F53B	TAB1	A	F858
TAB2	A	F89C	TAB3	A	F8C4	TABLE	A	F837	TBSE	A	FC4E
TCRET	A	F62F	TCSTS	A	F944	TG1	A	F829	TG2	A	F815
TG3	A	F7F9	THXB	A	F59B	THXN	A	F5A9	THXW	A	F604

TIMER	A	F95C	TPMR	A	FC51	TPMR1	A	FC53	TRUE	A	FFFF
TRUMP	A	FFFF	TSAV	A	0014	UCASE	A	0080	UM1	A	F9A7
UM2	A	F9AB	UM3	A	F9AF	UM4	A	F9B3	UM5	A	F9B7
UM6	A	FA5C	VDMCW	A	000B	VECTOR	A	FC55	VERY	A	FA9C
VERY1	A	FAA4	VERY2	A	FABF	VERY3	A	FAB5	WRI0	A	F2F0
WRI1	A	F305	WRI2	A	F308	WRI3	A	F323	WRITE	A	F2E8
X	A	F368	X0	A	F373	X1	A	F384	X2	A	F389
X3	A	F3A1	X4	A	F3C3	X5	A	F3C5	X6	A	F3D3
X7	A	F3D6	XTRACT	A	F681						

ASSEMBLY COMPLETE, NO ERRORS

HEX DUMP OF AMS80 REV 5.8


```
:D F000,FBFF
F000 C3 8B FC C3 CA F0 C3 DF F0 C3 8B F0 C3 A0 F0 C3
F010 B5 F0 C3 F4 F0 C3 24 F1 C3 28 F1 C3 52 F4 C3 77
F020 F4 C3 9C F1 C3 4D F1 C3 2D F1 C3 DF FB C3 44 F9
F030 C3 2F F6 C3 A9 F5 C3 9B F5 C3 04 F6 C3 48 F6 C3
F040 42 F4 C3 09 F1 C3 80 F5 C3 E6 F5 C3 3E F6 C3 14
F050 F6 C3 12 F1 C3 0C F6 C3 B1 F5 C3 25 F6 C3 1B F1
F060 C3 CF F5 C3 1C F6 C3 66 F0 C3 C7 F5 C3 F2 F5 C3
F070 38 F4 C3 5C F9 C3 FE F1 C3 48 FB C3 44 FB C3 C8
F080 FA C3 4D FB C3 84 F0 C3 87 F0 4F 3A 03 00 E6 03
F090 CA 2D FC FE 01 CA 2A FC FE 02 CA 27 FC C3 24 FC
F0A0 3A 03 00 E6 30 CA 09 FC FE 10 CA 06 FC FE 20 CA
F0B0 03 FC C3 00 FC 3A 03 00 E6 C0 CA 21 FC FE 40 CA
F0C0 1E FC FE 80 CA 1B FC C3 18 FC 3A 03 00 E6 03 CA
F0D0 39 FC FE 01 CA 36 FC FE 02 CA 33 FC C3 30 FC 3A
F0E0 03 00 E6 0C CA 15 FC FE 04 CA 12 FC FE 08 CA 0F
F0F0 FC C3 0C FC 3A 03 00 E6 03 CA 45 FC FE 01 CA 42
F100 FC FE 02 CA 3F FC C3 3C FC F5 C5 48 CD 8B F0 C1
F110 F1 C9 F5 C5 48 CD A0 F0 C1 F1 C9 F5 C5 48 CD B5
F120 F0 C1 F1 C9 3A 03 00 C9 79 32 03 00 C9 CD CA F0
F130 E6 7F C5 CD 8A F0 79 C1 C9 0C 41 4D 53 38 30 20
F140 52 45 56 2E 20 35 2E 38 2E 30 0D 0A FF 2A 51 FC
F150 25 7E 2F 77 BE 2F 77 C2 50 F1 23 06 12 11 41 F5
F160 1B 1A 2B 77 05 C2 60 F1 F9 01 EC FF 09 E5 21 00
F170 00 E5 E5 E5 3E C3 32 00 00 32 08 00 21 00 F0 22
F180 01 00 21 77 F4 22 09 00 3A 50 FC 32 03 00 3E 02
F190 CD 44 FB CD 2F F6 21 39 F1 CD 48 F6 CD 2F F6 0E
F1A0 3A CD 8B F0 CD 2D F1 D6 40 FA 9C F1 FE 1B D2 9C
F1B0 F1 0E 02 21 55 FC 87 85 6F 7E 23 66 6F E9 CD 62
F1C0 F4 11 F8 FF 19 F9 CD 2F F6 0E 3F CD 8B F0 C3 9F
F1D0 F1 CD 04 F4 CD 2F F6 D1 E1 E5 19 CD 04 F6 D5 0E
F1E0 20 CD 8B F0 D1 E1 E5 D5 CD FE F1 CD 04 F6 E1 D1
F1F0 CD FE F1 0E 20 CD 8B F0 CD 04 F6 C3 9C F1 7D 93
F200 6F 7C 9A 67 C9 0C CD 04 F4 C1 D1 E1 7E 02 03 CD
F210 38 F4 D2 0C F2 C3 9C F1 0C CD 04 F4 C1 D1 E1 71
F220 CD 38 F4 D2 1F F2 C3 9C F1 CD 04 F4 D1 E1 CD 2F
F230 F6 CD 04 F6 0E 20 CD 8B F0 7E CD 9B F5 CD 38 F4
F240 DA 9C F1 7D E6 0F C2 34 F2 CD 44 F9 C3 2E F2 0D
F250 CD 04 F4 CD 3E F6 0E 3A CD A0 F0 AF 57 CD 80 F5
F260 E1 CD 14 F6 3E 01 CD 80 F5 AF 92 CD 80 F5 CD E6
F270 F5 C3 9C F1 0D CD 04 F4 E1 E5 CD 1C F6 DE 3A C2
F280 78 F2 57 CD CF F5 CA A9 F2 5F CD CF F5 F5 CD CF
F290 F5 C1 4F 09 CD CF F5 CD CF F5 77 23 1D C2 97 F2
F2A0 CD CF F5 C2 BE F1 C3 78 F2 CD CF F5 67 CD CF F5
F2B0 6F B4 CA B6 F2 E9 E1 C3 9C F1 0D CD 04 F4 CD 45
F2C0 F4 DA BE F1 E1 7E CD 9B F5 0E 2D CD 8B F0 CD 42
F2D0 F4 DA 9C F1 CA E4 F2 E5 CD 30 F4 D1 E1 73 78 FE
F2E0 0D CA 9C F1 23 C3 C5 F2 CD 04 F4 CD 2F F6 D1 E1
F2F0 7D C6 10 4F 7C CE 00 47 7B 91 4F 7A 98 DA 05 F3
F300 3E 10 C3 08 F3 79 C6 11 B7 CA 9C F1 D5 5F 16 00
F310 CD 3E F6 0E 3A CD A0 F0 7B CD 80 F5 CD 14 F6 AF
F320 CD 80 F5 7E 23 CD 80 F5 1D C2 23 F3 AF 92 CD 80
F330 F5 D1 C3 F0 F2 CD 04 F4 D1 E1 CD 44 F9 CD 2F F6
F340 CD 04 F6 0E 20 CD 8B F0 7E FE FF CC 65 F3 FE 20
F350 FC 65 F3 CD 8A F0 CD 38 F4 DA 9C F1 7D E6 0F C2
F360 48 F3 C3 3A F3 3E 20 C9 CD 2D F1 21 41 F5 FE 0D
F370 CA D3 F3 BE CA 84 F3 F5 7E B7 FA BE F1 23 23 23
F380 F1 C3 73 F3 0E 20 CD 8B F0 23 7E EB 6F 26 00 39
F390 EB 23 46 23 1A CD 9B F5 05 CA A1 F3 1B 1A CD 9B
F3A0 F5 04 0E 2D CD 8B F0 CD 42 F4 DA 9C F1 CA C5 F3
```

xx = DEFAULT INITIALIZED INTERRUPT CONTROL MASK

```
F3B0 E5 C5 CD 30 F4 E1 F1 C5 F5 7D 12 C1 05 CA C3 F3
F3C0 13 7C 12 C1 E1 7E B7 FA 9C F1 78 FE 0D CA 9C F1
F3D0 C3 89 F3 CD 2F F6 0E 20 CD 8B F0 7E 23 B7 FA 9C
F3E0 F1 4F CD 8B F0 0E 3D CD 8B F0 7E 23 EB 6F 26 00
F3F0 39 EB 46 23 1A CD 9B F5 05 CA D6 F3 1B 1A CD 9B
F400 F5 C3 D6 F3 21 00 00 CD E5 FB 47 CD F2 F5 DA 1A
F410 F4 29 29 29 29 B5 6F C3 07 F4 E3 E5 78 CD 45 F4
F420 D2 28 F4 0D C2 BE F1 C9 C2 BE F1 0D C2 04 F4 C9
F430 0E 01 21 00 00 C3 0A F4 23 7C B5 37 C8 7B 95 7A
F440 9C C9 CD 2D F1 FE 20 C8 FE 2C C8 FE 0D 37 3F C0
F450 37 C9 E5 CD 5B F4 44 3E C0 E1 C9 C5 2A 53 FC C3
F460 66 F4 C5 2A 51 FC 25 7E 2F 77 BE 2F 77 C2 66 F4
F470 23 01 EE FF 09 C1 C9 E5 D5 C5 F5 CD 62 F4 EB 21
F480 0A 00 39 06 04 EB 2B 72 2B 73 D1 05 C2 86 F4 C1
F490 0B F9 21 14 00 39 7E 91 23 C2 A1 F4 7E 90 CA AF
F4A0 F4 23 23 7E 91 C2 AE F4 23 7E 90 CA AF F4 03 21
F4B0 0F 00 39 73 23 72 23 23 71 23 70 C5 CD 2F F6 0E
F4C0 25 CD 8B F0 E1 CD 04 F6 21 14 00 39 16 02 4E 36
F4D0 00 23 46 36 00 23 79 B0 CA DD F4 7E 02 23 15 C2
F4E0 CE F4 C3 9C F1 CD 42 F4 DA 27 F5 CA FF F4 CD 30
F4F0 F4 D1 21 13 00 39 72 2B 73 78 FE 0D CA 27 F5 16
F500 02 21 14 00 39 E5 0E 01 CD 04 F4 58 C1 E1 78 B1
F510 CA 1D F5 71 23 70 23 0A 77 23 3E CF 02 7B FE 0D
F520 CA 27 F5 15 C2 05 F5 CD 2F F6 21 08 00 39 E9 D1
F530 C1 F1 E1 F9 00 21 00 00 C3 00 00 00 00 00 00 00
F540 00 41 05 01 42 03 01 43 02 01 44 01 01 45 00 01
F550 46 04 01 48 10 01 4C 0F 01 4D 10 02 50 13 02 53
F560 07 02 FF CD 2F F6 0E 3D CD 8B F0 CD 2D F1 FE 0D
F570 CA 7A F5 4F CD A0 F0 C3 6B F5 CD 3E F6 C3 9C F1
F580 F5 0F 0F 0F 0F E6 0F CD C7 F5 CD A0 F0 F1 F5 E6
F590 0F CD C7 F5 CD A0 F0 F1 82 57 C9 F5 0F 0F 0F 0F
F5A0 E6 0F CD C7 F5 CD 8B F0 F1 E6 0F CD C7 F5 C3 8B
F5B0 F0 F5 0F 0F 0F 0F E6 0F CD C7 F5 CD B5 F0 F1 E6
F5C0 0F CD C7 F5 C3 B5 F0 C6 90 27 CE 40 27 4F C9 CD
F5D0 1C F6 CD F2 F5 07 07 07 07 4F CD 1C F6 CD F2 F5
F5E0 B1 4F 82 57 79 C9 06 3C 0E 00 CD A0 F0 05 C2 E8
F5F0 F5 C9 D6 30 D8 C6 E9 D8 C6 06 F2 00 F6 C6 07 D8
F600 C6 0A B7 C9 7C CD 9B F5 7D C3 9B F5 7C CD B1 F5
F610 7D C3 B1 F5 7C CD 80 F5 7D C3 80 F5 CD DF F0 DA
F620 BE F1 E6 7F C9 0E 0D CD B5 F0 0E 0A C3 B5 F0 0E
F630 0D CD 8B F0 0E 0D CD 8B F0 0E 0A C3 8B F0 0E 0D
F640 CD A0 F0 0E 0A C3 A0 F0 7E FE FF C8 CD 8A F0 23
F650 C3 48 F6 E1 E3 7E 23 E3 E5 C9 3C E6 07 FE 06 DA
F660 64 F6 C6 03 FE 05 DA 6B F6 C6 02 C6 41 4F C3 8B
F670 F0 06 04 4E CD 8B F0 23 05 C2 73 F6 0E 20 C3 8B
F680 F0 7A E6 38 0F 0F 0F C9 CD 81 F6 87 4F 21 34 F9
F690 09 4E CD 8B F0 23 4E CD 8B F0 0E 20 CD 8B F0 C3
F6A0 8B F0 CD 81 F6 E6 06 FE 06 C2 5A F6 0E 53 CD 8B
F6B0 F0 0E 50 C3 8B F0 0D CD 04 F4 CD 2F F6 E1 CD 44
F6C0 F9 E5 CD 04 F6 0E 20 CD 8B F0 CD 8B F0 CD 53 F6
F6D0 57 21 37 F8 01 11 00 BE CA 29 F8 23 0D C2 D7 F6
F6E0 0E 0A BE CA 15 F8 23 0D C2 E2 F6 0E 06 BE CA F9
F6F0 F7 23 0D C2 ED F6 E6 C0 FE 40 CA DF F7 FE 80 CA
F700 D0 F7 7A E6 C7 D6 04 CA C1 F7 3D CA BB F7 3D CA
F710 A7 F7 7A E6 C0 CA 7B F7 7A E6 C7 D6 C0 CA 70 F7
F720 D6 02 CA 65 F7 D6 02 CA 5A F7 D6 03 CA 4B F7 7A
F730 E6 07 4F 21 2B F9 09 CD 71 F6 CD 81 F6 FE 06 C2
F740 CA F7 21 28 F9 CD 71 F6 C3 BA F6 21 24 F9 CD 71
F750 F6 CD 81 F6 CD 9B F5 C3 BA F6 0E 43 CD 8B F0 CD
F760 88 F6 C3 04 F8 0E 4A CD 8B F0 CD 88 F6 C3 04 F8
```

```
F770 0E 52 CD 8B F0 CD 88 F6 C3 BA F6 21 0C F9 7A E6
F780 0F 3D CA 99 F7 FE 04 DA 8C F7 D6 05 87 87 4F 09
F790 CD 71 F6 CD A2 F6 C3 BA F6 CD 71 F6 CD A2 F6 0E
F7A0 2C CD 8B F0 C3 04 F8 21 08 F9 CD 71 F6 CD 81 F6
F7B0 CD 5A F6 0E 2C CD 8B F0 C3 20 F8 21 04 F9 C3 C4
F7C0 F7 21 00 F9 CD 71 F6 CD 81 F6 CD 5A F6 C3 BA F6
F7D0 7A E6 38 0F 4F 21 E0 F8 09 CD 71 F6 C3 F0 F7 21
F7E0 DC F8 CD 71 F6 CD 81 F6 CD 5A F6 0E 2C CD 8B F0
F7F0 7A E6 07 CD 5A F6 C3 BA F6 79 87 87 4F 21 C0 F8
F800 09 CD 71 F6 CD 53 F6 57 CD 53 F6 CD 9B F5 7A CD
F810 9B F5 C3 BA F6 79 87 87 4F 21 98 F8 09 CD 71 F6
F820 CD 53 F6 CD 9B F5 C3 BA F6 79 87 87 4F 21 54 F8
F830 09 CD 71 F6 C3 BA F6 00 07 0F 17 1F 27 2F 37 3F
F840 76 C9 E3 E9 EB F3 F9 FB C6 CE D3 D6 DB DE E6 EE
F850 F6 FE 22 2A 32 3A C3 CD 45 49 20 20 53 50 48 4C
F860 44 49 20 20 58 43 48 47 50 43 48 4C 58 54 48 4C
F870 52 45 54 20 48 4C 54 20 43 4D 43 20 53 54 43 20
F880 43 4D 41 20 44 41 41 20 52 41 52 20 52 41 4C 20
F890 52 52 43 20 52 4C 43 20 4E 4F 50 20 43 50 49 20
F8A0 4F 52 49 20 58 52 49 20 41 4E 49 20 53 42 49 20
F8B0 49 4E 20 20 53 55 49 20 4F 55 54 20 41 43 49 20
F8C0 41 44 49 20 43 41 4C 4C 4A 4D 50 20 4C 44 41 20
F8D0 53 54 41 20 4C 48 4C 44 53 48 4C 44 4D 4F 56 20
F8E0 41 44 44 20 41 44 43 20 53 55 42 20 53 42 42 20
F8F0 41 4E 41 20 58 52 41 20 4F 52 41 20 43 4D 50 20
F900 49 4E 52 20 44 43 52 20 4D 56 49 20 4C 58 49 20
F910 53 54 41 58 49 4E 58 20 44 41 44 20 4C 44 41 58
F920 44 43 58 20 52 53 54 20 50 53 57 20 50 4F 50 20
F930 50 55 53 48 4E 5A 5A 20 4E 43 43 20 50 4F 50 45
F940 50 20 4D 20 CD F4 F0 FE 00 C8 33 33 C3 9C F1 CD
F950 DF F0 4F CD A0 F0 CD 44 F9 C3 4F F9 1D C2 5C F9
F960 15 C2 5C F9 C9 CD 2F F6 21 A7 F9 CD 48 F6 3A 03
F970 00 CD C3 F9 21 AB F9 CD 48 F6 3A 03 00 CD BD F9
F980 21 AF F9 CD 48 F6 3A 03 00 CD BB F9 21 B3 F9 CD
F990 48 F6 3A 03 00 CD C1 F9 21 B7 F9 CD 48 F6 3A 0B
F9A0 00 CD C1 F9 C3 9C F1 20 43 3D FF 20 52 3D FF 20
F9B0 50 3D FF 20 4C 3D FF 20 56 3D FF 0F 0F 0F 0F 0F
F9C0 0F 07 07 E6 03 C3 9B F5 CD 2D F1 F5 0E 20 CD 8B
F9D0 F0 F1 FE 56 CA 2F FA FE 4C CA 1E FA FE 50 CA 0B
F9E0 FA FE 52 CA FA F9 FE 43 C2 BE F1 3A 03 00 CD C3
F9F0 F9 CD 48 FA 47 3E FC C3 5F FA 3A 03 00 CD BD F9
FA00 CD 48 FA 07 07 47 3E F3 C3 5F FA 3A 03 00 CD BB
FA10 F9 CD 48 FA 07 07 07 07 47 3E CF C3 5F FA 3A 03
FA20 00 CD C1 F9 CD 48 FA 0F 0F 47 3E 3F C3 5F FA 3A
FA30 0B 00 07 07 CD C3 F9 CD 48 FA 0F 0F 47 21 0B 00
FA40 7E E6 3F B0 77 C3 9C F1 21 5C FA CD 48 F6 CD 2D
FA50 F1 D6 30 FA BE F1 FE 04 F2 BE F1 C9 2D 20 FF 21
FA60 03 00 A6 B0 77 C3 9C F1 0D CD 04 F4 21 5C FA CD
FA70 48 F6 C1 21 05 00 36 DB 23 71 23 36 C9 CD 05 00
FA80 CD 9B F5 C3 9C F1 CD 04 F4 C1 D1 21 05 00 36 D3
FA90 23 73 23 36 C9 79 CD 05 00 C3 9C F1 CD 04 F4 D1
FAA0 E1 CD 2F F6 CD 44 F9 46 3E 55 77 BE C2 BF FA 2F
FAB0 77 BE C2 BF FA 70 CD 38 F4 DA 9C F1 C3 A4 FA CD
FAC0 04 F6 CD 2F F6 C3 B5 FA C5 D5 E5 FE 1B CA ED FA
FAD0 FE 1F CA F2 FA 11 03 00 CD 48 FB 4F 3E 20 A6 B1
FAE0 01 00 00 4F 21 04 FB 09 7E B7 C3 00 FB 0E 20 C3
FAF0 F4 FA 0E 00 11 03 00 CD 48 FB 3E DF A6 B1 77 37
FB00 E1 D1 C1 C9 00 45 0A 41 20 53 49 55 0D 44 52 4A
FB10 4E 46 43 4B 54 5A 4C 57 48 59 50 51 4F 42 47 00
FB20 4D 58 56 00 00 33 0A 2D 20 07 38 37 0D 24 34 27
```

```
FB30 2C 21 3A 28 35 22 29 32 1B 36 30 31 39 3F 26 FF
FB40 2E 2F 3B FF D3 DF FB C9 2A 0E 00 19 C9 C5 D5 E5
FB50 79 FE 20 CA 65 FB FE 0A CA 6A FB FE 0D C2 6F FB
FB60 3E 08 C3 B3 FB 3E 04 C3 B3 FB 3E 02 C3 B3 FB FE
FB70 60 DA 76 FB D6 20 4F 06 3F 21 43 FB 7E B9 CA 8D
FB80 FB 05 CA 89 FB 2B C3 7C FB 37 C3 B3 FB FE 40 D2
FB90 97 FB 0E 40 C3 99 FB 0E 80 11 03 00 CD 48 FB 3E
FBA0 C0 A6 57 7E E6 3F B1 77 7A B9 C2 AF FB 0E 00 78
FBB0 E6 1F B1 E1 D1 C1 C9 0D CD 04 F4 E1 CD 2F F6 0E
FBC0 3D CD 8B F0 CD 2D F1 FE 0D CA D6 FB FE 08 CA DB
FBD0 FB 77 23 C3 C4 FB 36 FF C3 9C F1 2B C3 C4 FB C5
FBE0 CD 8A F0 C1 C9 CD 2D F1 FE 20 CA E5 FB C9 F5 3E
FBF0 02 CD 44 FB F1 C3 E5 F4 F3 C3 00 E8 FF FF FF FF
```

D3 xx — WHERE xx IS YOUR INTERRUPT CONTROL PORT

00 EB = N* DOS BOOTSTRAP ENTRY

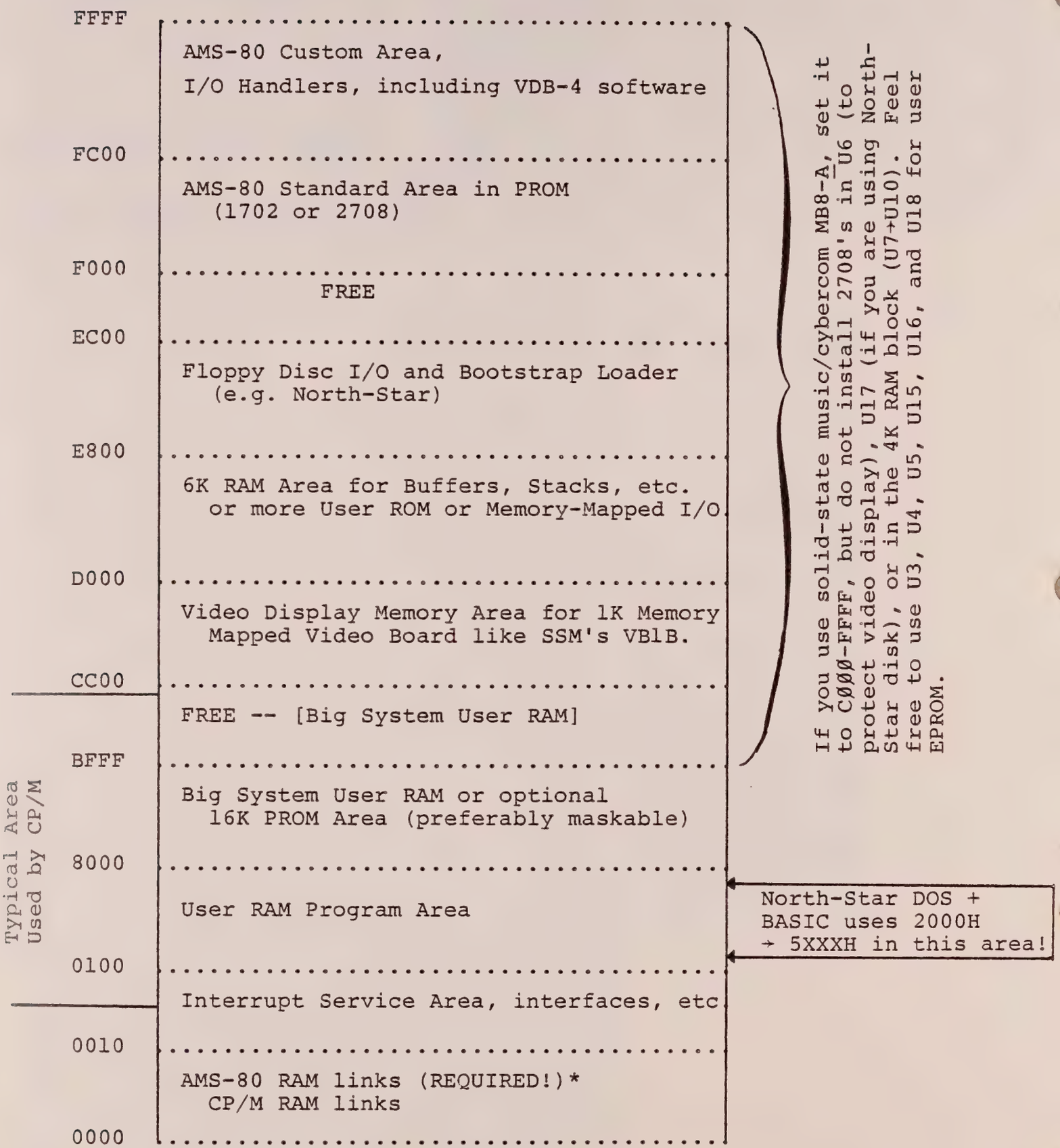
xx = @-SIGN'S DEFAULT INTERRUPT CONTROL MASK

ASCII "TEXT" IN AMS80 REV 5.8:

```
:A F120,F15F
F120     :   ъ2
F130
 ъ   AMS80
F140 REV. 5.8.0   *Q
F150 %"/w /w P $   A
```

```
:A F850,F94F
F850    "*2:  EI  SPHL
F860 DI  XCHGPCHLXTHL
F870 RET HLT CMC STC
F880 CMA DAA RAR RAL
F890 RRC RLC NOP CPI
F8A0 ORI XRI ANI SBI
F8B0 IN  SUI OUT ACI
F8C0 ADI CALLJMP LDA
F8D0 STA LHLDSHLDMOV
F8E0 ADD ADC SUB SBB
F8F0 ANA XRA ORA CMP
F900 INR DCR MVI LXI
F910 STAXINX DAD LDAX
F920 DCX RST PSW POP
F930 PUSHNZZ NCC POPE
F940 P M       33
```

TYPICAL AMS-80 SYSTEM MEMORY MAP


FFFF
AMS-80 Custom Area,
I/O Handlers, including VDB-4 software
FC00
AMS-80 Standard Area in PROM
(1702 or 2708)
F000
FREE
EC00
Floppy Disc I/O and Bootstrap Loader
(e.g. North-Star)
E800
6K RAM Area for Buffers, Stacks, etc.
or more User ROM or Memory-Mapped I/O
D000
Video Display Memory Area for 1K Memory
Mapped Video Board like SSM's VB1B.
CC00
FREE -- [Big System User RAM]
BFFF
Big System User RAM or optional
16K PROM Area (preferably maskable)
8000
User RAM Program Area
0100
Interrupt Service Area, interfaces, etc
0010
AMS-80 RAM links (REQUIRED!)*
CP/M RAM links
0000
Typical Area Used by CP/M
If you use solid-state music/cybercom MB8-A, set it to CØØØ-FFFF, but do not install 2708's in U6 (to protect video display), U17 (if you are using North-Star disk), or in the 4K RAM block (U7→U10). Feel free to use U3, U4, U5, U15, U16, and U18 for user EPROM.
North-Star DOS + BASIC uses 2000H → 5XXXH in this area!

*NOTE: Bytes ØØØE & F point to an additional RAM area used by AMS-80 and your software for tables, utility storage, etc. AMS-80 currently uses the first four (4) bytes of this secondary sto table which must be RAM, but may be located anywhere.

TYPICAL SYSTEM I/O ALLOCATION (CONFIGURATION)

PUNCH

P=3 → RAM*
P=2
P=1 audio cassette
P=0 console

READER

R=3 → RAM*
R=2
R=1 audio cassette
R=0 console

Selection of devices is governed by I/O CNTL byte at memory location 0003H.

7 6	5 4	3 2	1 0
LIST OUTPUT MSB	PUNCH OUTPUT	HI-SPEED READER INPUT	CONSOLE I/O LSB

LIST

L=3 → RAM*
L=2 Baudot device (model 15)
L=1 teletype port
L=0 console

CONSOLE IN/OUT

C=3 → RAM* FOR DRIVER TESTING
C=2 Baudot device (model 15)
C=1 video display board/keyboard
C=0 teletype port

VIDEO DISPLAY BOARD

V=0 Page with line foldover
V=1 Page without line foldover
V=2 Scroll with line foldover
V=3 Scroll without line foldover

Selection of video mode and options controlled by word 000B4 as documented in VDB-4 listing in Appendix V.

*You may allocate a RAM area for driver development in the FCØØ customization area. See Appendix C for a sample.

RAM RESERVED WORDS REQUIRED BY AMS-80/5.8

Although AMS-80 is ROM based it does require a certain amount of RAM. These words are reserved for AMS-80 and should not be used by any custom software. The utilization of these words in low memory is as follows:

Words	are used
0,1,2	AMS-80 at initialization so that any subsequent operation of the reset switch will cause AMS-80 to be activated.
Word 3	is used to control the I/O configuration per INTEL's I/O byte convention shown on page B-2.
Word 4	is not used.
Words 5,6,7	are used by the I and O commands of AMS-80. This is a transient usage. AMS-80 will rewrite these bytes on I and O commands.
Words 8,9,A	are used as a software breadkpoint by AMS-80 by means of the RST1 instruction. It also allows an RST1 hardware interrupt to force a breadpoint.
Word B	is the mode control byte for the page/scroll VDB4 driver software, protected line count, etc. See section V for a detailed description of the usage of this word.
Words C,D	are not used.
Words E,F	contain the base address of a block of non-contiguous RAM located in high memory, and excluding the video display area. This block is reserved for flags and data buffers. At the address contained in [000E,F] is the address of the cursor pointer in the VDB4 video display board software driver (two bytes long). The next byte is used to store the character currently being hidden beneath the cursor, The fourth byte is used by the Baudot TTY routines to save the current FIGS/LTRS shift case. Your software may use this buffer area if it protects the first four bytes pointed to by [000E,F].
Words 10-CBFF	are not used by AMS-80.
Words CCØØ-CFFF	are video RAM.

The following is reprinted from CP/M System Alteration Guide, page 18, (CP/M and this guide are © Digital Research, 1976 and 1977) and is included here FOR REFERENCE ONLY!

CP/M RESERVED WORDS AND AMS-80 COMPATIBILITY

Locations from	to	
ØØØØH	ØØØ2H	Contains a jump instruction to the warm start entry point at location 3E03H+b. This allows a simple programmed restart (JMP ØØØØH) or manual restart from the front panel. Similar to AMS-80 usage. AMS-80 and CP/M each will each rewrite this address when they take control.
ØØØ3H	ØØØ3H	Contains the INTEL standard IOBYTE, which is optionally included in the user's CBIOS, as described in Section 6. Exact compatibility with AMS-80!
ØØØ4H	ØØØ4H	(not currently used - reserved) some versions of CP/M use this byte to flag ID of currently active disk drive.
ØØØ5H	ØØØ7H	Contains a jump instruction to the BDOS, and serves two purposes: JMP ØØØ5H provides the primary entry point to the BDOS, as described in the manual "CP/M Interface Guide," and LHLD ØØØ6H brings the address field of the instruction to the HL register pair. This value is the lowest address in memory used by CP/M (assuming the CCP is being overlayed). Note that the DDT program will change the address field to reflect the reduced memory size in debug mode.
ØØØ8H	ØØ27H	(interrupt locations 1 through 5 not used) AMS-80 uses and initializes ØØØ8-ØØØF on Entry.
ØØ3ØH	ØØ37H	(interrupt location 6, not currently used - reserved)
ØØ38H	ØØ3AH	Contains a jump instruction into the DDT program when running in debug mode for programmed breakpoints, but is not otherwise used by CP/M.
ØØ3BH	ØØ3FH	(not currently used - reserved)
ØØ4ØH	ØØ4FH	16 byte area reserved for scratch by CBIOS, but is not used for any purpose in the distribution version of CP/M. Many of you have I/O jump tables in the FCØØ area that point here for RAM development of I/O drivers. Use with some caution!

No conflicts with AMS-80!

ØØ5ØH - ØØ5BH	(not currently used - reserved)
ØØ5CH - ØØ7CH	default file control block produced for a transient program by the Console Command Processor.
ØØ7DH - ØØ7FH	(not currently used - reserved)
ØØ8ØH - ØØFFH	default 128 byte disk buffer (also filled with the command line when a transient is loaded under the CCP).

Note that this information is set-up for normal operation under the CP/M system, but can be overwritten by a transient program if the BDOS facilities are not required by the transient.

AMS-80's I/O entry points in the jump table at FØØØ are "perfectly" compatible with CP/M's BDOS I/O requirements since both use INTEL's "INTELLEC" and MDS I/O conventions.

```
N* DOS REV 50 / AMS80 REV 5.8
+GO ASMB

ASMB DEVELOPMENT SYSTEM
F /CUSTM/ 6000
CUSTM 6000  6900

R
FILE
FC00DIST,2
CUSTM 6000  7A34

A FC00
```

UNLESS WE HAVE DISCUSSED SPECIFIC CHANGES, THE FOLLOWING SET OF I/O UTILITIES ARE "BURNED" INTO YOUR EPROMS. UNFORTUNATELY, I CAN'T SECOND-GUESS YOUR SYSTEM REQUIREMENTS! THE "HINTS & KINKS" IN SECTION (H) MAY GIVE YOU SOME IDEAS ON THINGS TO DO TO MEET YOUR OWN REQUIREMENTS.

DE W3IWI

```
FC00                0010 ;-- INTERFACES TO AMS80/5.8 ----
FC00                0020 ;
FC00                0030 ;-- SKELETON CUSTOMIZATION FOR (SAMPLE)...
FC00                0040 ;
FC00                0050 ; THIS VERSION PREPARED BY T.A.CLARK ...27 AUG 79
FC00                0060 ;
FC00                0070 ; NOTE THAT THIS ASSEMBLER ONLY USES FIVE LETTER MNEMONICS
FC00                0080 ; ALTHOUGH IT DOESN'T BOUNCE ON LONGER NAMES. THEREFORE,
FC00                0090 ; READ A VARIABLE LIKE "TTYOUT" TO BE THE SAME AS "TTYOU".
FC00                0100 ;
FC00                0110 ;-----------------------------------------
FC00                0120 ;
FC00                0130 ;--VARIOUS ENTRY POINTS AND DEFINITIONS:
FC00                0140 ;
FC00                0150 AMS80  EQU   0F000H        ;START OF AMS80
FC00                0160 BEGIN  EQU   AMS80+14DH    ;ENTRY POINT AFTER INITIALIZATION
FC00                0170 CONSO  EQU   AMS80+08BH    ;CONSOLE OUTPUT ENTRY POINT
FC00                0180 CONSI  EQU   AMS80+0CAH    ;CONSOLE INPUT ENTRY POINT
FC00                0190 ERROR  EQU   AMS80+1BEH    ;COMMAND ERROR RE-ENTRY POINT
FC00                0200 IOROM  EQU   0FC00H        ;I/O & CUSTOM ROM ADDRESS
FC00                0210 VDB4   EQU   0FD00H        ;VIDEO DISPLAY DRIVER
FC00                0220 VDRAM  EQU   0CC00H        ;AMS80 DEFINITION OF VIDEO RAM AREA
FC00                0230 UTRAM  EQU   0D000H        ;UTILITY RAM BLOCK ADDRESS
FC00                0240 RBADR  EQU   000EH         ;POINTER TO UTRAM IN BASE-PAGE
FC00                0250 IORAM  EQU   40H           ;RAM AREA FOR I/O DRIVER DEVELOPMENT
FC00                0260 ;
FC00                0270 ; I CHOOSE TO DEFAULT VDB4 TO PAGE MODE, WITH LINE TRUNCATION
FC00                0280 ; AND WITH ONE PROTECTED LINE AT THE TOP OF THE SCREEN:
FC00                0290 VDSET  EQU   01H           ;DEFAULT VIDEO MODE
FC00                0300 ;
FC00                0310 VDMCW  EQU   000BH         ;<VIDEO CONTROL BYTE ADDR>
FC00                0320 ;
FC00                0330 ; LACKING FURTHER INFO, I'LL CHOOSE TO MAKE AMS80
FC00                0340 ; WAKE UP FOR YOU WITH C,L,P & R ALL = 0 :
FC00                0350 IODEF  EQU   0             ;DEFAULT I/O CONTROL BYTE
FC00                0360 ;
FC00                0370 ; TTY-LIKE DEVICE DEFINITION CONSTANTS:
FC00                0380 TTSTA  EQU   0             ;TTY STATUS PORT
FC00                0390 TTDAT  EQU   1             ;TTY DATA PORT
FC00                0400 INFLG  EQU   40H           ;TTY INPUT STATUS FLAG
FC00                0410 OUTFL  EQU   80H           ;TTY OUTPUT STATUS FLAG
FC00                0420 ;
FC00                0430 ; LACKING FURTHER INFO, I WILL ASSUME THAT YOU WANT
FC00                0440 ; TO BYPASS OVERRUN AND FRAMING ERRORS:
FC00                0450 ERRFL  EQU   00H           ;TTY FRAMING/OVERRUN ERROR FLAG
```

```
FC00                    0460 ;
FC00                    0470 ; LACKING FURTHER INFO, I WILL ASSUME THAT YOU DON'T
FC00                    0480 ; HAVE A DEDICATED NON-TTY KEYBOARD BY DEFAULTING IT TO
FC00                    0490 ; YOUR TTY-LIKE INPUT ROUTINE, BUT I'LL LEAVE ROOM SO THAT
FC00                    0500 ; YOU CAN CODE IN A KEYBOARD AT A LATER TIME:
FC00                    0510 KBSTA  EQU   TTSTAT
FC00                    0520 KBDAT  EQU   TTDATA
FC00                    0530 KBFLG  EQU   INFLG
FC00                    0540 ;
FC00                    0550 ; LACKING FURTHER INFO ON YOUR MEMORY CONFIGURATION, I'LL
FC00                    0560 ; ASSUME THAT YOU WANT YOUR STACK AND USER TOP OF MEMORY
FC00                    0570 ; LOCATED IMMEDIATELY BELOW THE VIDEO RAM AREA:
FC00                    0580 TOP    EQU   VIRAM+255
FC00                    0590 TOP1   EQU   VIRAM+255
FC00                    0600 ;
FC00                    0610 ;-----------------------------------
FC00                    0620 ;-- I/O CUSTOMIZING TABLE --
FC00                    0630 ;
FC00                    0640         ORG   IOROM
FC00                    0650 ;  MUST BEGIN AT FC00H ! ! ! !
FC00                    0660 ;-- PUNCH DEVICES
FC00 C3 40 00           0670 PUN3   JMP   IORAM
FC03 C3 00 FD           0680 PUN2   JMP   VDB4
FC06 C3 B4 FC           0690 PUN1   JMP   TTYOUT
FC09 C3 EB F0           0700 PUN0   JMP   CONSO
FC0C                    0710 ;-- READER DEVICES
FC0C C3 43 00           0720 RDR3   JMP   IORAM+3
FC0F C3 CA F0           0730 RDR2   JMP   CONSI
FC12 C3 CA F0           0740 RDR1   JMP   CONSI
FC15 C3 CA F0           0750 RDR0   JMP   CONSI
FC18                    0760 ;-- LIST DEVICES
FC18 C3 46 00           0770 LIST3  JMP   IORAM+6
FC1B C3 00 FD           0780 LIST2  JMP   VDB4
FC1E C3 B4 FC           0790 LIST1  JMP   TTYOUT
FC21 C3 EB F0           0800 LIST0  JMP   CONSO
FC24                    0810 ;-- CONSOLE OUTPUT DEVICES
FC24 C3 49 00           0820 COUT3  JMP   IORAM+9
FC27 C3 00 FD           0830 COUT2  JMP   VDB4
FC2A C3 B4 FC           0840 COUT1  JMP   TTYOUT
FC2D C3 00 FD           0850 COUT0  JMP   VDB4
FC30                    0860 ;-- CONSOLE INPUT DEVICES
FC30 C3 4C 00           0870 CIN3   JMP   IORAM+12
FC33 C3 CE FC           0880 CIN2   JMP   KBDIN
FC36 C3 99 FC           0890 CIN1   JMP   TTYIN
FC39 C3 99 FC           0900 CIN0   JMP   TTYIN
FC3C                    0910 ;-- CONSOLE KEYBOARD STATUS
FC3C C3 4F 00           0920 CST3   JMP   IORAM+15
FC3F C3 E0 FC           0930 CST2   JMP   KBDST
FC42 C3 BF FC           0940 CST1   JMP   TTYST
FC45 C3 BF FC           0950 CST0   JMP   TTYST
FC48                    0960 ;-- BAUDOT I/O ENTRIES
FC48 C3 52 00           0970 BDTTI  JMP   IORAM+18
FC4B C3 55 00           0980 BDTTO  JMP   IORAM+21
FC4E                    0990 ;-----------------------
FC4E                    1000 ;-- MEMORY DEFINITION CONSTANTS
FC4E                    1010 ;  MUST BEGIN AT FC4EH ! ! ! !
FC4E                    1020 ; DEFINE UTILITY RAM BUFFER AREA:
FC4E 00 D0              1030 TBGE   DW    UTRAM
FC50                    1040 ; DEFINE DEFAULT I/O STATE:
FC50 00                 1050 IORW   DB    IODEF
```

C-3

```
FC51                    1070 ; DEFINE TOP OF USER RAM AREAS:
FC51 FF CC              1080 TMEMR  DW    TOP
FC53 FF CC              1090 TMEM1  DW    TOP1
FC55                    1100 ;
FC55                    1110 ;-------------------------------
FC55                    1120 ;-- COMMAND JUMP TABLE
FC55                    1130 ;  MUST BEGIN AT FC55H ! ! ! !
FC55                    1140 VECT   EQU   $             ;COMMAND TABLE
FC55 EE FB              1150        DW    AMS80+0BEEH   ;AT-SIGN
FC57 35 F3              1160        DW    AMS80+335H    ;A
FC59 00 2A              1170        DW    2A00H         ;B=JUMP TO LOADED N* BASIC ETC (SET TO 2DØØH FOR
FC5B C8 F9              1180        DW    AMS80+9C8H    ;C                              DOUBLE-DENSITY N*
FC5D 29 F2              1190        DW    AMS80+229H    ;D                              SYSTEMS)
FC5F 4F F2              1200        DW    AMS80+24FH    ;E
FC61 18 F2              1210        DW    AMS80+218H    ;F
FC63 E5 F4              1220        DW    AMS80+4E5H    ;G
FC65 D1 F1              1230        DW    AMS80+1D1H    ;H
FC67 68 FA              1240        DW    AMS80+0A68H   ;I
FC69 B7 FB              1250        DW    AMS80+0BB7H   ;J
FC6B BE F1              1260        DW    ERROR         ;K
FC6D B6 F6              1270        DW    AMS80+6B6H    ;L
FC6F 05 F2              1280        DW    AMS80+205H    ;M
FC71 6E F2              1290        DW    AMS80+26EH    ;N
FC73 86 FA              1300        DW    AMS80+0A86H   ;O
FC75 4F F9              1310        DW    AMS80+94FH    ;P
FC77 F8 FB              1320        DW    AMS80+0BF8H   ;Q
FC79 74 F2              1330        DW    AMS80+274H    ;R
FC7B BA F2              1340        DW    AMS80+2BAH    ;S
FC7D 63 F5              1350        DW    AMS80+563H    ;T
FC7F 65 F9              1360        DW    AMS80+965H    ;U
FC81 9C FA              1370        DW    AMS80+0A9CH   ;V
FC83 EB F2              1380        DW    AMS80+2EBH    ;W
FC85 68 F3              1390        DW    AMS80+368H    ;X
FC87 BE F1              1400        DW    ERROR         ;Y
FC89 00 29              1410        DW    2900H         ;Z=JUMP TO LOADED N* DOS
FC8B                    1420 ;-------------------------------
FC8B                    1430 ;
FC8B                    1440 ;-- INITIALIZATION SECTION --
FC8B                    1450 ;  MUST BEGIN AT FC8BH ! ! ! !
FC8B                    1460 ;
FC8B 3E 01              1470 INIT   MVI   A,VIDSET    ;SET UP VDB4 TO
FC8D 32 0B 00           1480        STA   VDMCW       ;  MEET YOUR DESIRES
FC90 2A 4E FC           1490        LHLD  TBGE        ;INITIALIZE UTILITY RAM
FC93 22 0E 00           1500        SHLD  RBADR       ;  BUFFER ADDRESS
FC96 C3 4D F1           1510        JMP   BEGIN       ;AND GO TO THE WONDERFUL WORLD OF AMS80!
FC99                    1520 ;  OTHER CUSTOMIZED INITIALIZATION ROUTINES SHOULD
FC99                    1530 ;  BE INCLUDED JUST BEFORE THE "JMP BEGIN" EXIT POINT.
FC99                    1540 ;
FC99                    1550 ;--------- SAMPLE I/O ROUTINES ----------
FC99                    1560 ;  THESE SAMPLE ROUTINES JUST FILL THE FC00-FCFF ROM AREA.
FC99                    1570 ;  IF YOU REQUIRE ADDITIONAL CUSTOMIZATION OR ADDITIONAL I/O
FC99                    1580 ;  ROUTINES, I WOULD SUGGEST THAT YOU RE-LOCATE ALL I/O
FC99                    1590 ;  ROUTINES INTO THE FF00-FFFF ROM AREA TO KEEP THEM TOGETHER.
FC99                    1600 ;--TTY INPUT INTO <A> REGISTER
FC99 CD ED FC           1610 TTYIN  CALL  HOLDST
FC9C C2 AD FC           1620        JNZ   TTYI2
FC9F DB 00              1630 TTYI1  IN    TTSTAT
FCA1 E6 40              1640        ANI   INFLG
FCA3 CA 9F FC           1650        JZ    TTYI1
FCA6 DB 00              1660        IN    TTSTAT
```

```
FCA9 E6 00              1670        ANI   ERRFLG
FCAA C2 B2 FC           1680        JNZ   BAD
FCAD DB 01              1690 TTYI2  IN    TTDATA
FCAF C3 F3 FC           1700        JMP   MSKCLR
FCB2 37                 1710 BAD    STC
FCB3 C9                 1720        RET
FCB4                    1730 ;---TTY OUTPUT, DATA FROM <C>, CLOBBER <A>
FCB4 DB 00              1740 TTYOU  IN    TTSTAT
FCB6 E6 80              1750        ANI   OUTFLG
FCB8 CA B4 FC           1760        JZ    TTYOUT
FCBB 79                 1770        MOV   A,C
FCBC D3 01              1780        OUT   TTDATA
FCBE C9                 1790        RET
FCBF                    1800 ;--- TTY STATUS
FCBF CD ED FC           1810 TTYST  CALL  HOLDST
FCC2 C2 C9 FC           1820        JNZ   RSTAT
FCC5 DB 00              1830        IN    TTSTAT
FCC7 E6 40              1840        ANI   INFLG
FCC9                    1850 ;--RETURN <A>=FF IF DATA AVAILABLE, 0 IF NOT:
FCC9 3E 00              1860 RSTAT  MVI   A,0
FCCB C8                 1870 NODAT  RZ                ;A=0
FCCC 3D                 1880 AVAIL  DCR   A           ;A=-1
FCCD C9                 1890        RET
FCCE                    1900 ;--- DEDICATED KEYBOARD INPUT
FCCE CD ED FC           1910 KBDIN  CALL  HOLDST
FCD1 C2 DB FC           1920        JNZ   KBDI2
FCD4 DB 00              1930 KBDI1  IN    KBSTAT
FCD6 E6 40              1940        ANI   KBFLG
FCD8 CA D4 FC           1950        JZ    KBDI1
FCDB DB 01              1960 KBDI2  IN    KBDATA
FCDD C3 F3 FC           1970        JMP   MSKCLR
FCE0                    1980 ;--- KEYBOARD STATUS
FCE0 CD ED FC           1990 KBDST  CALL  HOLDST
FCE3 C2 C9 FC           2000        JNZ   RSTAT
FCE6 DB 00              2010        IN    KBSTAT
FCE8 E6 40              2020        ANI   KBFLG
FCEA C3 C9 FC           2030        JMP   RSTAT
FCED                    2040 ;-- THIS ROUTINE LOOKS AT THE 20H BIT
FCED                    2050 ;   IN VDMCW TO SEE IF THE CURRENT
FCED                    2060 ;   INPUT REGISTER HAS DATA IN HOLD:
FCED 3A 0B 00           2070 HOLDS  LDA   VDMCW
FCF0 E6 20              2080        ANI   20H           ;MASK HOLD BIT
FCF2 C9                 2090        RET
FCF3                    2100 ;--THIS ROUTINE MASKS OFF PARITY BIT AND
FCF3                    2110 ;  CLEARS OUT THE CHARACTER-IN-HOLD BIT:
FCF3 E6 7F              2120 MSKCL  ANI   7FH           ;STRIP OFF PARITY
FCF5 F5                 2130        PUSH  PSW           ;TEMP SAVE
FCF6 3A 0B 00           2140        LDA   VDMCW         ;SET UP TO
FCF9 E6 DF              2150        ANI   0FFH-20H      ;CLEAR OUT
FCFB 32 0B 00           2160        STA   VDMCW         ;HOLD BIT
FCFE F1                 2170        POP   PSW           ;RESTORE DATA
FCFF C9                 2180        RET
FD00                    2190 ZEND   EQU   $             ;NEXT AVAIL ADDR
FD00                    2200        END
```

ASSUMES THAT DATA READY SETS KBFLG BIT HIGH WHEN KEY HAS BEEN STRUCK. SEE W81W1 CUSTOMIZATION LINES 3380-3480 FOR KBFLG = LOW ON VALID DATA CODE.

SYMBOL TABLE

```
AMSB0 F000   AVAIL FCCC   BAD   FCB2   BDTTI FC48   BDTTO FC4B
BEGIN F14D   CIN0  FC39   CIN1  FC36   CIN2  FC33   CIN3  FC30
CONSI F0CA   CONSO F08B   COUT0 FC2D   COUT1 FC2A   COUT2 FC27
```

```
COUT3 FC24   CST0  FC45   CST1  FC42   CST2  FC3F   CST3  FC3C
ERRFL 0000   ERROR F1BE   HOLDG FCED   INFLG 0040   INIT  FC8B
IODEF 0000   IORAM 0040   IOROM FC00   IORW  FC50   KBDAT 0001
KBDI1 FCD4   KBDI2 FCDB   KBDIN FCDE   KBDST FCE0   KBFLG 0040
KBSTA 0000   LIST0 FC21   LIST1 FC1E   LIST2 FC1B   LIST3 FC18
MSKCL FCF3   NODAT FCCB   OUTFL 0069   PUN0  FC09   PUN1  FC06
PUN2  FC03   PUN3  FC00   RBADR 000E   RDR0  FC15   RDR1  FC12
RDR2  FC0F   RDR3  FC0C   RSTAT FCC9   T8GE  FC4E   TMEM1 FC53
TMEMR FC51   TOP   CCFF   TOP1  CCFF   TTDAT 0001   TTSTA 0000
TTYI1 FC9F   TTYI2 FCAD   TTYIN FC99   TTYOU FCB4   TTYST FCBF
UTRAM D000   VDB4  FD00   VDMOU 000B   VDRAM CC00   VDSET 0001
VECT  FC55   ZEND  FD00
```

THE FOLLOWING PAGES CONTAIN THE CODE FOR THE CUSTOM AREA CURRENTLY IN USE AT W3IWI. THEY ARE INCLUDED TO GIVE YOU SOME IDEAS ON OTHER "TRICKS" YOU CAN DO WITH YOUR SYSTEM. ALSO REFER TO THE "HINTS & KINKS" SECTION FOR MORE IDEAS.

SAMPLE CUSTOMIZATION #2
W3IWI'S PERSONAL SYSTEM

C-6

A FC00

```
FC00                    0010 ;--- INTERFACES TO AMS80/5.8 -----
FC00                    0020 ;
FC00                    0030 ;--- W3IWI CUSTOMIZATION 2 JULY 1979
FC00                    0040 ;--- REVISED TO INCLUDE TRUMP BOARD 24 AUG 79
FC00                    0050 ;-----------------------------------------
FC00                    0060 ;
FC00                    0070 ;---VARIOUS ENTRY POINTS AND DEFINITIONS:
FC00                    0080 ;
FC00                    0090 AMS80   EQU   0F000H        ;START OF AMS80
FC00                    0100 BEGIN   EQU   AMS80+14DH    ;ENTRY POINT AFTER INITIALIZATION
FC00                    0110 CONSO   EQU   AMS80+08BH    ;CONSOLE OUTPUT ENTRY POINT
FC00                    0120 CONSI   EQU   AMS80+0CAH    ;CONSOLE INPUT ENTRY POINT
FC00                    0130 ERROR   EQU   AMS80+1BEH    ;COMMAND ERROR RE-ENTRY POINT
FC00                    0140 CUSTM   EQU   0FC00H        ;CUSTOM ROM ADDRESS
FC00                    0150 IOROM   EQU   0FF00H        ;I/O DRIVERS
FC00                    0160 VDB4    EQU   0FD00H        ;VIDEO DISPLAY DRIVER
FC00                    0170 VDRAM   EQU   0CC00H        ;AMS80 STANDARD VIDEO RAM AREA
FC00                    0180 UTRAM   EQU   0D000H        ;UTILITY RAM BLOCK ADDRESS
FC00                    0190 RBADR   EQU   000EH         ;POINTER TO UTRAM IN BASE-PAGE
FC00                    0200 IORAM   EQU   0D800H        ;RAM FOR I/O DRIVER DEVELOPMENT
FC00                    0210 ;
FC00                    0220 ;DEFAULT VDB4 TO PAGE MODE, WITH LINE TRUNCATION
FC00                    0230 ; AND WITH ONE PROTECTED LINE AT THE TOP OF THE SCREEN:
FC00                    0240 VDSET   EQU   01H           ;DEFAULT VIDEO MODE
FC00                    0250 IOCW    EQU   0003H         ;I/O CONTROL BYTE
FC00                    0260 ;
FC00                    0270 VDMCW   EQU   000BH         ;<VIDEO CONTROL BYTE ADDR>
FC00                    0280 ;
FC00                    0290 ;---DEFAULT TO 2 FOR AUTO CONSOLE SELECTION:
FC00                    0300 IODEF   EQU   2             ;DEFAULT I/O CONTROL BYTE
FC00                    0310 ;
FC00                    0320 ; TTY-LIKE DEVICE DEFINITION CONSTANTS:
FC00                    0330 ;
FC00                    0340 ; SERIAL PORT A DEFINITIONS:
FC00                    0350 ;
FC00                    0360 ASTAT   EQU   0             ;STATUS PORT
FC00                    0370 ADATA   EQU   1             ;DATA PORT
FC00                    0380 AIFLG   EQU   40H           ;INPUT STATUS FLAG
FC00                    0390 AOFLG   EQU   80H           ;OUTPUT STATUS FLAG
FC00                    0400 AEFLG   EQU   0             ;FRAMING/OVERRUN ERROR FLAG
FC00                    0410 ;
FC00                    0420 ; DITTO, SERIAL PORT B:
FC00                    0430 BSTAT   EQU   2
FC00                    0440 BDATA   EQU   3
FC00                    0450 BIFLG   EQU   40H
FC00                    0460 BOFLG   EQU   80H
FC00                    0470 BEFLG   EQU   0
FC00                    0480 ;
FC00                    0490 ; PARALLEL INPUT KEYBOARD
FC00                    0500 ;
FC00                    0510 KSTAT   EQU   BSTAT         ;USES A SER-B BIT
FC00                    0520 KDATA   EQU   5             ;PAR-B INPUT PORT (LATCHED)
FC00                    0530 KBFLG   EQU   1             ;=LSB IN BSTAT
FC00                    0540 ;       (INTEGRAL DATA SYSTEMS ID-225)
FC00                    0550 ; I.D.S. PRINTER INTERFACES:
FC00                    0560 ;
FC00                    0570 PSTAT   EQU   8             ;SER DATA PORT ON 3P+S
FC00                    0580 PDATA   EQU   9             ;STATUS PORT ON 3P+S
```

```
FC00                    0590 ;  TX BFR =80H, CTS FROM PRINTER =01H:
FC00                    0600 POFLG  EQU   81H          ;READY TO SEND FLAG
FC00                    0610 POMSK  EQU   80H          ;MASK FOR NEXT CHAR OK
FC00                    0620 ;
FC00                    0630 ;--- FRONT PANEL SWITCH FOR PRINTER CONTROL
FC00                    0640 ;
FC00                    0650 PANEL  EQU   0FFH         ;SWITCH REGISTER (ALTAIR/IMSAI"STANDARD")
FC00                    0660 CECHO  EQU   02H          ;CONSOLE PRINTER ECHO MASK
FC00                    0670 PRTON  EQU   01H          ;PRINTER ON/OFF MASK
FC00                    0680 ;
FC00                    0690 ;--- TRUMP INITIALIZATION CONSTANTS:
FC00                    0700 ;
FC00                    0710 TRUMP  EQU   0B0H         ;BASE ADR PER MAGNANIMOUS AGREEMENT
                                                       ;OF AMSAT'S "HARDCORE"!
FC00                    0720 P8253  EQU   TRUMP+0      ;8253 COUNTERS
FC00                    0730 CNTR0  EQU   P8253        ;COUNTER#0
FC00                    0740 CNTR1  EQU   P8253+1      ;  #1
FC00                    0750 CNTR2  EQU   P8253+2      ;  #2
FC00                    0760 CNTRC  EQU   P8253+3      ; CONTROL PORT
FC00                    0770 ;
FC00                    0780 P8255  EQU   TRUMP+4      ;8255 PARALLEL PORT BASE ADR.
FC00                    0790 ;
FC00                    0800 P8251  EQU   TRUMP+8      ;8251 SERIAL PORT BASE ADR.
FC00                    0810 ;
FC00                    0820 I8214  EQU   TRUMP+10     ;8214 INT CNTL PORT
FC00                    0830 ;
FC00                    0840 ;--- TRUMP STROBES WITH IN/OUT:
FC00                    0850 ;
FC00                    0860 ARST   EQU   TRUMP+11     ;ASYNC RESET STROBE
FC00                    0870 CLOCK  EQU   TRUMP+14     ;50 HZ RT CLOCK
FC00                    0880 TONE   EQU   TRUMP+15     ;800 HZ TONE ON/OFF
FC00                    0890 ;
FC00                    0900 ; STACK AND USER RAM AREA DEFINITIONS:
FC00                    0910 ;
FC00                    0920 TOP    EQU   0E0FFH       ;FIND TOP RAM FOR AM680 STACK.
FC00                    0930 TOP1   EQU   VDRAM+255
FC00                    0940 ;
FC00                    0950 ;-------------------------------------
FC00                    0960 ;--- I/O CUSTOMIZING TABLE ----
FC00                    0970 ;
FC00                    0980        ORG   CUSTM
FC00                    0990 ;  MUST BEGIN AT FC00H ! ! ! !
FC00                    1000 ;--- PUNCH DEVICES
FC00 C3 00 D8           1010 PUN3   JMP   IORAM
FC03 C3 45 FF           1020 PUN2   JMP   BOUT
FC06 C3 D0 FF           1030 PUN1   JMP   PRINT
FC09 C3 8B F0           1040 PUN0   JMP   CONSO
FC0C                    1050 ;--- READER DEVICES
FC0C C3 03 D8           1060 RDR3   JMP   IORAM+3
FC0F C3 12 FF           1070 RDR2   JMP   BIN
FC12 C3 00 FF           1080 RDR1   JMP   AIN
FC15 C3 CA F0           1090 RDR0   JMP   CONSI
FC18                    1100 ;--- LIST DEVICES
FC18 C3 06 D8           1110 LIST3  JMP   IORAM+6
FC18 C3 45 FF           1120 LIST2  JMP   BOUT
FC1E C3 3A FF           1130 LIST1  JMP   AOUT
FC21 C3 D0 FF           1140 LIST0  JMP   PRINT
FC24                    1150 ;--- CONSOLE OUTPUT DEVICES
FC24 C3 09 D8           1160 COUT3  JMP   IORAM+9
FC27 CD 00 FD           1170 COUT2  CALL  VDB4         ;BOTH ON UNTIL AUTO SELECTION !
```

WHEN MY SYSTEM "WAKES UP" IT LOOKS AT BOTH MY PARALLEL INPUT KEYBOARD & SERIAL INPUT (ADM-3) WAITING FOR A KEY TO BE STRUCK. WHICH EVER COMES FIRST BECOMES THE SYSTEM CONSOLE. THIS IS DONE BY FORCING CONSOLE #2 TO BE A "POLL FOR INPUT" UTILITY, WHICH DECIDES IF CONSOLE #0 OR #1 IS TO BE ASSIGNED. AT ANY TIME I CAN SELECT CONSOLE #2 TO FORCE A RE-SELECTION. W3IWI

```
FC2A C3 33 FF            1180 COUT1  JMP   AOUTX
FC2D C3 53 FF            1190 COUT0  JMP   VDB4X
FC30                     1200 ;--- CONSOLE INPUT DEVICES
FC30 C3 0C D8            1210 CIN3   JMP   IORAM+12
FC33 C3 2D FF            1220 CIN2   JMP   SWIN          ;AUTO SELECT WILL ASSIGN 0 OR 1
FC36 C3 00 FF            1230 CIN1   JMP   AIN
FC39 C3 9C FF            1240 CIN0   JMP   KBDIN
FC3C                     1250 ;--- CONSOLE KEYBOARD STATUS
FC3C C3 0F D8            1260 CST3   JMP   IORAM+15
FC3F C3 5D FF            1270 CST2   JMP   SWIST         ;WAIT FOR AUTO SELECTION
FC42 C3 80 FF            1280 CST1   JMP   AIST
FC45 C3 AE FF            1290 CST0   JMP   KBST
FC48                     1300 ;--- BAUDOT I/O ENTRIES
FC48 C3 12 D8            1310 BDTTI  JMP   IORAM+18
FC4B C3 15 D8            1320 BDTTO  JMP   IORAM+21
FC4E                     1330 ;
FC4E                     1340 ;-------------------------
FC4E                     1350 ;--- MEMORY DEFINITION CONSTANTS
FC4E                     1360 ;  MUST BEGIN AT FC4EH ! ! ! !
FC4E                     1370 ;
FC4E                     1380 ; DEFINE UTILITY RAM BUFFER AREA:
FC4E 00 D0               1390 TBSE   DW    UTRAM
FC50                     1400 ; DEFINE DEFAULT I/O STATE:
FC50 02                  1410 IORW   DB    IODEF
FC51                     1420 ; DEFINE TOP OF USER RAM AREAS:
FC51 FF E0               1430 TMEMR  DW    TOP
FC53 FF CC               1440 TMEM1  DW    TOP1
FC55                     1450 ;
FC55                     1460 ;--------------------------------
FC55                     1470 ;--- COMMAND JUMP TABLE
FC55                     1480 ;  MUST BEGIN AT FC55H ! ! ! !
FC55                     1490 ;
FC55                     1500 VECT   EQU   $             ;COMMAND TABLE
FC55 EE FB               1510         DW    AMS80+0BEEH  ;AT-SIGN
FC57 35 F3               1520         DW    AMS80+335H   ;A
FC59 00 2D               1530         DW    2D00H        ;B=JUMP TO LOADED N* BASIC ETC
FC5B C8 F9               1540         DW    AMS80+9C8H   ;C          (DOUBLE DENSITY)
FC5D 29 F2               1550         DW    AMS80+229H   ;D
FC5F 4F F2               1560         DW    AMS80+24FH   ;E
FC61 18 F2               1570         DW    AMS80+218H   ;F
FC63 E5 F4               1580         DW    AMS80+4E5H   ;G
FC65 D1 F1               1590         DW    AMS80+1D1H   ;H
FC67 68 FA               1600         DW    AMS80+0A68H  ;I
FC69 B7 FB               1610         DW    AMS80+0BB7H  ;J
FC6B BE F1               1620         DW    ERROR        ;K
FC6D B6 F6               1630         DW    AMS80+6B6H   ;L
FC6F 05 F2               1640         DW    AMS80+205H   ;M
FC71 6E F2               1650         DW    AMS80+26EH   ;N
FC73 86 FA               1660         DW    AMS80+0A86H  ;O
FC75 4F F9               1670         DW    AMS80+94FH   ;P
FC77 F8 FB               1680         DW    AMS80+0BF8H  ;Q
FC79 74 F2               1690         DW    AMS80+274H   ;R
FC7B BA F2               1700         DW    AMS80+2BAH   ;S
FC7D 63 F5               1710         DW    AMS80+563H   ;T
FC7F 65 F9               1720         DW    AMS80+965H   ;U
FC81 9C FA               1730         DW    AMS80+0A9CH  ;V
FC83 E8 F2               1740         DW    AMS80+2E8H   ;W
FC85 68 F3               1750         DW    AMS80+368H   ;X
FC87 03 2A               1760         DW    2A03H        ;Y=JUMP TO ALREADY LOADED IPS
FC89 00 29               1770         DW    2900H        ;Z=JUMP TO LOADED N* DOS
```

CONSOLE DEVICE #2 FOR AUTO SELECT!

```
FC8B                    1780 ;--------------------------------
FC8B                    1790 ;
FC8B                    1800 ;--- INITIALIZATION SECTION ---
FC8B                    1810 ;  MUST BEGIN AT FC8BH ! ! ! !
FC8B                    1820 ;
FC8B 3E 01              1830 INIT   MVI   A,VDSET       ;SET UP VDB4 TO
FC8D 32 0B 00           1840        STA   VDMCW         ;  MEET YOUR DESIRES
FC90 2A 4E FC           1850        LHLD  TBSE          ;INITIALIZE UTILITY RAM
FC93 22 0E 00           1860        SHLD  RBADR         ;  BUFFER ADDRESS
FC96 3E B7              1870        MVI   A,0B7H        ;COUNTER#2 ON 8253 ON
FC98                    1880 ;                          TRUMP BOARD - LOAD BCD INIT.
FC98                    1890 ;--- TRUMP BOARD INITIALIZATION:
FC98                    1900 ;
FC98 D3 B3              1910        OUT   CNTRC         ;LOAD INIT
FC9A 3E 00              1920        MVI   A,000H        ;FOR 50
FC9C D3 B2              1930        OUT   CNTR2         ; HERTZ
FC9E 3E 25              1940        MVI   A,025H        ;  CLOCK
FCA0 D3 B2              1950        OUT   CNTR2         ;   TICKS
FCA2                    1960 ; (ADD 8251 & 8255 INITIALIZATION HERE)
FCA2 C3 4D F1           1970        JMP   BEGIN         ;THE WONDERFUL WORLD OF AMS80!
FCA5                    1980 ;
FCA5                    1990 ;--- VDB4 BELL VIA TRUMP:
FCA5                    2000 ;
FCA5                    2010        ORG   CUSTM+0F0H
FCF0 D5                 2020 DING   PUSH  D
FCF1 11 00 20           2030        LXI   D,2000H
FCF4 D3 BF              2040        OUT   TONE          ;TURN IT ON
FCF6 CD 72 F0           2050        CALL  0F072H        ;AMS80 DELAY UTILITY
FCF9 F5                 2060        PUSH  PSW
FCFA DB BF              2070        IN    TONE          ;TURN IT OFF (DUMMY DATA)
FCFC F1                 2080        POP   PSW
FCFD D1                 2090        POP   D
FCFE C9                 2100        RET
FCFF                    2110 ;
FCFF                    2120 ;--- PATCH DING INTO VDB4:
FCFF                    2130 ;
FCFF                    2140        ORG   VDB4+83H
FD83 CD F0 FC           2150        CALL  DING
FD86                    2160 ;
FD86                    2170 ;--- PATCH TRUMP INTERRUPT INTO AMS80:
FD86                    2180 ;
FD86                    2190        ORG   AMS80+0B44H
FB44 D3 BA              2200        OUT   I8214
FB46                    2210 ;
FB46                    2220 ; ------------------------------------------------
FB46                    2230 ;
FB46                    2240 ;---SERIAL-A INPUT, RETURN DATA IN <A>
FB46                    2250 ;
FB46                    2260        ORG   IOROM
FF00 CD BC FF           2270 AIN    CALL  HOLDST
FF03 C2 0D FF           2280        JNZ   AIN2
FF06 DB 00              2290 AIN1   IN    ASTAT
FF08 E6 40              2300        ANI   AIFLG
FF0A CA 06 FF           2310        JZ    AIN1
FF0D DB 01              2320 AIN2   IN    ADATA
FF0F C3 C2 FF           2330        JMP   MSKCLR
FF12                    2340 ;
FF12                    2350 ;---SERIAL-B INPUT, JUST LIKE AIN:
FF12                    2360 ;
FF12 CD BC FF           2370 BIN    CALL  HOLDST
```

```
FF15 C2 28 FF            2380         JNZ   BIN2
FF18 DB 02               2390 BIN1    IN    BSTAT
FF1A E6 40               2400         ANI   BIFLG
FF1C CA 18 FF            2410         JZ    BIN1
FF1F DB 02               2420         IN    BSTAT
FF21 E6 00               2430         ANI   BEFLG
FF23 CA 28 FF            2440         JZ    BIN2
FF26 37                  2450 BAD     STC
FF27 C9                  2460         RET
FF28 DB 03               2470 BIN2    IN    BDATA
FF2A C3 C2 FF            2480         JMP   MSKCLR
FF2D                     2490 ;
FF2D                     2500 ;--- AUTOMATIC SELECTION OF INPUT CONSOLE OPTION.
FF2D                     2510 ;  SETTING TO CONSOLE #2 WILL WAIT FOR INPUT FROM
FF2D                     2520 ;  EITHER KEYBOARD AND RESET IOCW ACCORDINGLY:
FF2D                     2530 ;
FF2D CD 5D FF            2540 SWIN    CALL  SWIST
FF30 C3 CA F0            2550         JMP   CONSI          ;NOW PRINT CHARACTER
FF33                     2560 ;
FF33                     2570 ;
FF33                     2580 ;---SER-A OUTPUT, DATA FROM <C>, CLOBBER <A>
FF33                     2590 ;--- AOUTX ENTRY PERMITS CONSOLE ECHO ON PRINTER:
FF33                     2600 ;
FF33 DB FF               2610 AOUTX   IN    PANEL
FF35 E6 02               2620         ANI   CECHO
FF37 C4 D0 FF            2630         CNZ   PRINT
FF3A DB 00               2640 AOUT    IN    ASTAT
FF3C E6 80               2650         ANI   AOFLG
FF3E CA 3A FF            2660         JZ    AOUT
FF41 79                  2670 AOUT1   MOV   A,C
FF42 D3 01               2680         OUT   ADATA
FF44 C9                  2690         RET
FF45                     2700 ;
FF45                     2710 ;---DITTO, SERIAL-B WITHOUT PRINTER ECHO OPTION
FF45                     2720 ;
FF45                     2730 ;ORG IOROM+50H
FF45 DB 02               2740 BOUT    IN    BSTAT
FF47 D2 00 FD            2750         JNC   VDB4
FF4A                     2760 ;
FF4A E6 80               2770         ANI   BOFLG
FF4C CA 45 FF            2780         JZ    BOUT
FF4F 79                  2790 BOUT1   MOV   A,C
FF50 D3 03               2800         OUT   BDATA
FF52 C9                  2810         RET
FF53                     2820 ;
FF53                     2830 ;
FF53                     2840 ;--- VDB4 ECHO ENTRY TO PERMIT CONSOLE ECHO ON PRINTER
FF53                     2850 ;
FF53 DB FF               2860 VDB4X   IN    PANEL
FF55 E6 02               2870         ANI   CECHO
FF57 C4 D0 FF            2880         CNZ   PRINT
FF5A C3 00 FD            2890         JMP   VDB4
FF5D                     2900 ;
FF5D                     2910 ;--- SWIST WAITS FOR INPUT FROM EITHER KEYBOARD AND
FF5D                     2920 ;  DIDDLES IOCW TO RE-ASSIGN THE CONSOLE.
FF5D                     2930 ;
FF5D 3A 03 00            2940 SWIST   LDA   IOCW           ;GET OLD ONE
FF60 E6 FC               2950         ANI   0FCH           ;SAVE 6 MSB
FF62 4F                  2960         MOV   C,A            ;  AND TEMP SAVE
FF63 CD B4 FF            2970 SET0    CALL  KBST1          ;KEYBOARD INPUT?
```

#02 BIT ON PANEL INPUT REGISTER CONTROLS PRINTER ECHO OF ALL KEYBOARD INPUT

```
FF66 B7             2980        ORA   A
FF67 C2 71 FF       2990        JNZ   SET2
FF6A CD 86 FF       3000 SET1   CALL  AIST1          ;ADM-3?
FF6D 3C             3010        INR   A
FF6E C2 63 FF       3020        JNZ   SET0           ;LOOP FOR INPUT
FF71 3C             3030 SET2   INR   A
FF72 B1             3040        ORA   C
FF73 32 03 00       3050        STA   IOCW           ;RESET IOCW
FF76 3A 0B 00       3060        LDA   VDMCW
FF79 F6 20          3070        ORI   20H            ;SET CHAR IN HOLD BIT
FF7B 32 0B 00       3080        STA   VDMCW
FF7E 97             3090        SUB   A              ;RETURN 0 AS STATUS
FF7F C9             3100        RET
FF80                3110 ;
FF80                3120 ;--- SERIAL-A STATUS:
FF80                3130 ;
FF80 CD BC FF       3140 AIST   CALL  HOLDST
FF83 C2 8A FF       3150        JNZ   RSTAT
FF86 DB 00          3160 AIST1  IN    ASTAT
FF88 E6 40          3170        ANI   AIFLG
FF8A                3180 ;--RETURN <A>=FF IF DATA AVAILABLE, 0 IF NOT:
FF8A 3E 00          3190 RSTAT  MVI   A,0
FF8C C8             3200 NODAT  RZ                   ;A=0
FF8D 3D             3210 AVAIL  DCR   A              ;A=-1
FF8E C9             3220        RET
FF8F                3230 ;
FF8F                3240 ;---DITTO, SERIAL-B
FF8F                3250 ;
FF8F CD BC FF       3260 BIST   CALL  HOLDST
FF92 C2 8A FF       3270        JNZ   RSTAT
FF95 DB 02          3280 BIST1  IN    BSTAT
FF97 E6 40          3290        ANI   BIFLG
FF99 C3 8A FF       3300        JMP   RSTAT
FF9C                3310 ;
FF9C                3320 ;--- DEDICATED KEYBOARD INPUT ROUTINES:
FF9C                3330 ;
FF9C CD BC FF       3340 KBDIN  CALL  HOLDST
FF9F C2 A9 FF       3350        JNZ   KBDI2
FFA2 DB 02          3360 KBDI1  IN    KSTAT
FFA4 E6 01          3370        ANI   KBFLG
FFA6 C2 A2 FF       3380        JNZ   KBDI1
FFA9 DB 03          3390 KBDI2  IN    KDATA
FFAB C3 C2 FF       3400        JMP   MSKCLR
FFAE                3410 ;
FFAE                3420 ;--- KEYBOARD STATUS
FFAE                3430 ;
FFAE CD BC FF       3440 KBST   CALL  HOLDST
FFB1 C2 8A FF       3450        JNZ   RSTAT
FFB4 DB 02          3460 KBST1  IN    KSTAT
FFB6 2F             3470        CMA
FFB7 E6 01          3480        ANI   KBFLG
FFB9 C3 8A FF       3490        JMP   RSTAT
FFBC                3500 ;
FFBC                3510 ;-- THIS ROUTINE LOOKS AT THE 20H BIT
FFBC                3520 ;   IN VDMCW TO SEE IF THE CURRENT
FFBC                3530 ;   INPUT REGISTER HAS DATA IN HOLD:
FFBC                3540 ;
FFBC 3A 0B 00       3550 HOLDS  LDA   VDMCW
FFBF E6 20          3560        ANI   20H            ;MASK HOLD BIT
FFC1 C9             3570        RET
```

MY KEYBOARD SETS STATUS BIT LOW WHEN A KEY IS STRUCK.

```
FFC2                    3580 ;
FFC2                    3590 ;--THIS ROUTINE MASKS OFF PARITY BIT AND
FFC2                    3600 ;  CLEARS OUT THE CHARACTER-IN-HOLD BIT:
FFC2                    3610 ;
FFC2 E6 7F              3620 MSKCL  ANI   7FH           ;STRIP OFF PARITY
FFC4 F5                 3630        PUSH  PSW           ;TEMP SAVE
FFC5 3A 0B 00           3640        LDA   VDMCW         ;SET UP TO
FFC8 E6 DF              3650        ANI   0FFH-20H      ;CLEAR OUT
FFCA 32 0B 00           3660        STA   VDMCW         ;HOLD BIT
FFCD F1                 3670        POP   PSW           ;RESTORE DATA
FFCE C9                 3680        RET
FFCF                    3690 ;
FFCF                    3700 ;---PRINTER OUTPUT DRIVER
FFCF                    3710 ;
FFCF                    3720        ORG   IOROM+0D0H
FFD0 DB FF              3730 PRINT  IN    PANEL
FFD2 E6 01              3740        ANI   PRTON
FFD4 C8                 3750        RZ
FFD5 DB 08              3760 PRNT0  IN    PSTAT
FFD7 E6 81              3770        ANI   POFLG
FFD9 FE 80              3780        CPI   POMSK
FFDB C2 D0 FF           3790        JNZ   PRINT
FFDE 79                 3800 PRNT1  MOV   A,C
FFDF D3 09              3810        OUT   PDATA
FFE1 C9                 3820        RET
FFE2                    3830 ;
FFE2                    3840 ZEND   EQU   $             ;NEXT AVAIL ADDR
FFE2                    3850        END
```

#01 BIT ON PANEL TURNS PRINTER ON/OFF UNCONDITIONALLY

SYMBOL TABLE

```
ADATA 0001   AEFLG 0000   AIFLG 0040   AIN   FF00   AIN1  FF06
AIN2  FF0D   AIST  FF80   AIST1 FF86   AMS80 F000   AOFLG 0080
AOUT  FF3A   AOUT1 FF41   AOUTX FF33   ARST  00BB   ASTAT 0000
AVAIL FF8D   BAD   FF26   BDATA 0003   BDTTI FC48   BDTTO FC4B
BEFLG 0000   BEGIN F14D   BIFLG 0040   BIN   FF12   BIN1  FF18
BIN2  FF28   BIST  FF8F   BIST1 FF95   BOFLG 0080   BOUT  FF45
BOUT1 FF4F   BSTAT 0002   CECHO 0002   CIN0  FC39   CIN1  FC36
CIN2  FC33   CIN3  FC30   CLOCK 00BE   CNTR0 00B0   CNTR1 00B1
CNTR2 00B2   CNTRC 00B3   CONSI F0CA   CONSO F08B   COUT0 FC2D
COUT1 FC2A   COUT2 FC27   COUT3 FC24   CST0  FC45   CST1  FC42
CST2  FC3F   CST3  FC3C   CUSTM FC00   DING  FCF0   ERROR F1BE
HOLDS FFEC   I8214 00BA   INIT  FC88   IOCW  0003   IODEF 0002
IORAM D800   IOROM FF00   IORW  FC50   KBDI1 FFA2   KBDI2 FFA9
KBDIN FF9C   KBFLG 0001   KBST  FFAE   KBST1 FFB4   KDATA 0005
KSTAT 0002   LIST0 FC21   LIST1 FC1E   LIST2 FC1B   LIST3 FC18
MSKCL FFC2   NODAT FF8C   P8251 00B8   P8253 00B0   P8255 00B4
PANEL 00FF   PDATA 0009   POFLG 0081   POMSK 0080   PRINT FFD0
PRNT0 FFD5   PRNT1 FFDE   PRTON 0001   PSTAT 0008   PUN0  FC09
PUN1  FC06   PUN2  FC03   PUN3  FC00   RBADR 000E   RDR0  FC15
RDR1  FC12   RDR2  FC0F   RDR3  FC0C   RSTAT FF8A   SET0  FF63
SET1  FF6A   SET2  FF71   SWIN  FF2D   SWIST FF5D   TBSE  FC4E
THEM1 FC53   THEMR FC51   TONE  00BF   TOP   E0FF   TOP1  CCFF
TRUMP 00B0   UTRAM D000   VDB4  FD00   VDB4X FF53   VDMCW 000B
VDRAM CC00   VDSET 0001   VECT  FC55   ZEND  FFE2
```

HINTS AND KINKS:

THE FOLLOWING ALLOWS CONSOLE & LIST DEVICES TO BE BOTH CRT & HARD-COPY:

```
FC18                    0810 ;
FC18                    0820 ;--- LIST DEVICES
FC18 C3 46 00           0830 LIST3  JMP   IORAM+6
FC1B CD 20 FF           0840 LIST2  CALL  TTYOUT        ;TTY PLUS CRT
FC1E C3 00 FD           0850 LIST1  JMP   VDB4          ;CRT ONLY
FC21 C3 20 FF           0860 LIST0  JMP   TTYOUT        ;TTY ONLY
FC24                    0870 ;
FC24                    0880 ;--- CONSOLE OUTPUT DEVICES
FC24 C3 49 00           0890 COUT3  JMP   IORAM+9
FC27 CD 00 FD           0900 COUT2  CALL  VDB4          ;=CRT PLUS TTY
FC2A C3 20 FF           0910 COUT1  JMP   TTYOUT        ;TTY ONLY
FC2D C3 00 FD           0920 COUT0  JMP   VDB4          ;CRT ONLY
FC30                    0930 ;
```

THE CALL FOLLOWED BY JUMP EXECUTES BOTH ROUTINES (CONSOLE #2 OR LIST #2), WHILE THE #1 ENTRY EXECUTES ONLY ONE.

(DE W3IW1)

PROCESSOR TECHNOLOGY CUTS BOARD INTERFACES

```
        IF      CUTS
;THESE ROUTINES ARE COMPATABLE WITH CUTS BOARD
;BY PROC TECH
CASI:   ;CASSETTE INPUT DRIVER
        IN      CCP
        ANI     40H
        JZ      CASI    ;WAIT FOR IT
        IN      CCP
        ANI     8       ;TEST FOR FRAME ERROR
        JNZ     CBAD    ;ERROR
        IN      CDP     ;GO GET IT
        ANI     7FH     ;MASK PARITY BIT
        RET

CBAD:   STC             ;SET CARRY FLAG
        RET

CASO:   ;CASSETTE OUTPUT DRIVER
        IN      CCP
        ANI     80H
        JZ      CASO    ;WAIT FOR IT
        MOV     A,C
        OUT     CDP     ;DUMP IT
        RET
        ENDIF
```

(DE G3ZCZ)

BAUDOT OUTPUT DRIVER FOR AR-1

```
BAUD0:  CALL    ASCBD   ;CONVERT CHARACTER
        RC              ;RETURN FOR CONTROL CODE
                        ;RECOGNITION
        PUSH    PSW     ;SAVE CHARACTER
        RLC             ;TEST BIT7
        JC      BAUD1   ;BIT7 WAS SET
        RLC             ;TEST BIT6
        JNC     BAUD2   ;BIT6 WAS NOT SET
        MVI     C,FIGS  ;SET UP TO PRINT "FIGS"
        JMP     BAUD4
BAUD1:  MVI     C,LETS  ;SET UP TO PRINT "LETS"
BAUD4:  CALL    BAUD3   ;PRINT IT AND RETURN  HERE
BAUD2:  POP     PSW     ;RETRIEVE CHARACTER
        ANI     1FH     ;MASK FLAG BITS
        MOV     C,A     ;SET UP TO
BAUD3:  JMP     TTO     ;PRINT CHARACTER

FIGS    EQU     37Q
LETS    EQU     33Q
```

(DE G3ZCZ)

NON-LATCHING KEYBOARDS ARE NOT RECOMMENDED, BUT IF YOU MUST:

```
                        ;KEYBOARD INPUT ROUTINE
                        ;NON LATCHING KEYBOARD
KRFLG   EQU     80H     ;KEYBOARD STROBE BIT
KBD     EQU     4       ;KEYBOARD PORT
TIMER   EQU     0F072H
KEYIN:  IN      KBD     ;GET STATUS
        ANI     KRFLG   ;AND TEST IT
        JNZ     KEYIN   ;WAIT FOR IT
        CALL    DELAY   ;DEBOUNCE
        IN      KBD     ;NOW GET CHARACTER
        PUSH    PSW     ;AND SAVE IT
K1:     IN      KBD     ;WAIT FOR END OF STROBE
        ANI     KRFLG
        JZ      K1
        PUSH    D
        LXI     D, 54H  ; EXPERIMENT WITH DEBOUNCE TIME!
        CALL    TIMER   ; AMS80 ENTRY AT F072H
        POP     D
        POP     PSW     ;RETRIEVE CHARACTER
        CMA             ;SET IT RIGHT
        ANI     7FH     ;MASK IT
        RET
```

(DE G3ZCZ)

INITIALIZATION OF CUTS SERIAL PORT:

```
        IF      CUTS
LOW     EQU     20H     ;BIT 5 = 300 BAUD
CCP     EQU     0FAH    ;CASSETTE CONTROL PORT
CPD     EQU     0FBH    ;         DATA PORT
        MVI     A,LOW
        OUT     CCP     ;SET CASSETTE INTERFACE
        ENDIF
```

(DE G3ZCZ)

INITIALIZATION OF G3ZCZ's AR-1 "AMATEUR RADIO" BOARD FOR RTTY USE:

```
        IF      AR1
UCP     EQU     1       ;AR-1 CONTROL PORT
UDP     EQU     0       ;      DATA PORT
BCRPT   EQU     2       ;BAUD RATE PORT
        XRA     A       ;IN ORDER TO ALLOW RE-ENTRY
        OUT     UCP     ;AT "START" , IT IS
        OUT     UCP     ;NECESSARY TO
        OUT     UCP     ;DUMMY UP 8251
        MVI     A,40H   ;NOW RESET IT
        OUT     UCP
        MVI     A,82H   ;ASYNC 5 LEVEL MODE
                        ;[ USE 0EEH FOR 8 LEVEL ]
        OUT     UCP
        MVI     A,15H   ;COMMAND
        OUT     UCP
        MVI     A,30H   ;SET UP 45.5 BAUDS
        OUT     BCRPT
        ENDIF
```

(DE G3ZCZ)

BAUDOT RTTY INPUT DRIVER FOR AR-1:

```
BAUDI:  ;BAUDOT DEVICE INPUT ROUTINE
        CALL    BAUD6   ;FETCH CHAR
        RNC             ;RETURN IF VALID
                        ;ELSE ASSUME FIGS/LETTS
BAUD6:  CALL    TTI     ;GET SERIAL INPUT
        RC              ;ERROR INPUT
        ANI     1FH     ;MASK 5 LSB
        JMP     BACON   ;CONVERT AND RETURN

BAUDO:  ;BAUDOT DEVICE OUTPUT ROUTINE
        PUSH    PSW     ;SAVE <REG'S>
        PUSH    B       ;TO MAKE 'CHIN' WORK
        CALL    BAUDO
        POP     B
        POP     PSW
        RET
```

(DE G3ZCZ)

W⌀PN REPORTS THAT THE FOLLOWING IS REQUIRED TO MAKE AN 8251 WORK PROPERLY FOR CHARACTER OUTPUT (HE USES AN RO-33 AS A PRINTER):

```
                        1800 ;-----------------------------------------------------------
                        1810 ;
                        1820 ; 8251 UART-MODE INITIALIZATION:
                        1830 ;
FF00 3E 00              1840 I8251  MVI   A,0           ;BANG IT
FF02 D3 09              1850        OUT   TTYST         ; THRICE
FF04 D3 09              1860        OUT   TTYST         ;  TO MAKE IT
FF06 D3 09              1870        OUT   TTYST         ;   OH SO NICE !
FF08 3E 50              1880        MVI   A,RESET       ;NOW RESET IT
FF0A D3 09              1890        OUT   TTYST
FF0C 3E EE              1900        MVI   A,MDSET       ;AND SET MODE
FF0E D3 09              1910        OUT   TTYST
FF10 C3 4D F1           1920        JMP   BEGIN         ;AND GO TO THE WONDERFUL WORLD OF
                                                                              AMS80!
FF13                    1930 ;
FF13                    1940 ;---------- I/O ROUTINES -------------
FF13                    1950 ;
FF13                    1960        ORG   IOROM+20H
FF20                    1970 ;
FF20                    1980 ;---TTY OUTPUT, DATA FROM <C>, CLOBBER <A>
FF20 DB 09              1990 TTYOU  IN    TTYST
FF22 E6 01              2000        ANI   TXRDY
FF24 CA 20 FF           2010        JZ    TTYOUT
FF27                    2020 ;
FF27                    2030 ; FOLLOWING CODE IS PER W0PN AND W3IWI DISAVOWS ANY CREDIT
FF27                    2040 ; OR BLAME FOR SAME !!!!
FF27                    2050 ;
FF27 3E 33              2060        MVI   A,TXEN
FF29 D3 09              2070        OUT   TTYST
FF2B 79                 2080        MOV   A,C
FF2C D3 08              2090        OUT   TTDATA
FF2E DB 09              2100 RETRN  IN    TTYST
FF30 E6 04              2110        ANI   DUMB
FF32 CA 2E FF           2120        JZ    RETRN
FF35                    2130 ; FOLLOWING REVISION PER W0PN LETTER 18 MAR
FF35 3E 02              2140        MVI   A,2
FF37 D3 09              2150        OUT   TTYST
FF39 C9                 2160        RET
```

(DE W⌀PN)

ASR-37 AS A PAPER-TAPE READER

```
;THIS ROUTINE DRIVES AN IO-4 INPUT PORT

PTR:            ;PAPER TAPE READER
                ;MODEL 37 VIA PARALLEL PORT
        MVI     A,STROBE
        OUT     RCP     ;TURN RELAY ON
        IN      RSP     ;TEST STATUS
        ANI     STBT
        JNZ     PTR     ;WAIT FOR IT
PTR1:   MVI     A,40H
        DCR     A       ;DEBOUNCE
        JNZ     $-1
        IN      RDP     ;FETCH CHARACTER
        PUSH    PSW     ;SAVE IT
        MVI     A,CLEAR
        OUT     RCP     ;RESET RELAY
PTR4:   IN      RSP     ;TEST STATUS
        ANI     STBT
        JZ      PTR4    ;WAIT FOR CHARACTER TO
                        ;PASS HOLES
        POP     PSW     ;RETRIEVE IT
        RET
```

(DE G3ZCZ)

STATION CONTROLLER BOARD SOFTWARE

H-6

```
E000                    0010 ; DRIVER FOR AMSAT ANTENNA CONTROLLER BOARD'S REAL-TIME CLOC

E000                    0020 ;DE W3IWI      JUNE 1978
E000                    0030 ;
E000 C3 49 E0           0040 GETIM  JMP   GCLOK         ;TIME GETTER ENTRY
E003 C3 6A E0           0050 GPACK  JMP   PCLOK         ;PACKED MMM OR SSS
E006                    0060 ;--- CLOCK DRIVER FOR UPPER RIGHT CORNER OF CRT --
E006                    0070 CPORT  EQU   27H           ;I/O PORT FOR CLOCK
E006                    0080 CRTIM  EQU   0CC38H        ;UPPER RIGHT CORNER
E006                    0090 SECND  EQU   CRTIM+7
E006                    0100 MASK   EQU   192           ;ASCII CONV
E006                    0110 VDMCW  EQU   0BH           ;VDM CNTL BYTE
E006                    0120 TIMEB  EQU   8             ;VDMCW CLOCK CONTROL BIT
E006                    0130 ;--- CLOCK READ AND SET TIME INTO VIDEO RAM:
E006 F5                 0140 CLOCK  PUSH  PSW
E007 3A 0B 00           0150        LDA   VDMCW
E00A E6 08              0160        ANI   TIMEB
E00C C2 1C E0           0170        JNZ   RETRN         ;IF CLOCK OFF
E00F E5                 0180        PUSH  H
E010 DB 27              0190 CLOK1  IN    CPORT
E012 D6 C0              0200        SUI   MASK
E014 21 3F CC           0210        LXI   H,SECND
E017 BE                 0220        CMP   M
E018 C2 1E E0           0230        JNZ   UPDAT
E01B E1                 0240        POP   H
E01C F1                 0250 RETRN  POP   PSW
E01D C9                 0260        RET
E01E D5                 0270 UPDAT  PUSH  D
E01F 11 43 E0           0280        LXI   D,CTABL
E022 21 38 CC           0290        LXI   H,CRTIM
E025 CD 35 E0           0300        CALL  UTIM2
E028 CD 32 E0           0310        CALL  UTIM3
E02B CD 32 E0           0320        CALL  UTIM3
E02E D1                 0330        POP   D
E02F                    0340 ;LOOP BACK TO MAKE SURE IT DIDN'T CHANGE DURING READ:
E02F C3 10 E0           0350        JMP   CLOK1
E032 36 3A              0360 UTIM3  MVI   M,':'
E034 2C                 0370        INR   L
E035 CD 38 E0           0380 UTIM2  CALL  UTIM1
E038 1A                 0390 UTIM1  LDAX  D
E039 D3 27              0400        OUT   CPORT
E03B DB 27              0410        IN    CPORT
E03D D6 C0              0420        SUI   MASK
E03F 77                 0430        MOV   M,A
E040 2C                 0440        INR   L
E041 1C                 0450        INR   E
E042 C9                 0460        RET
E043 07                 0470 CTABL  DB    7             ;10HRS
E044 06                 0480        DB    6             ;HOURS
E045 05                 0490        DB    5             ;10MINS
E046 01                 0500        DB    1             ;MINS
E047 02                 0510        DB    2             ;10SEC
E048 03                 0520        DB    3             ;SECS
E049                    0530 ; GCLOK GETS CLOCK TIME FROM SCREEN
E049                    0540 ; CALL WITH <E> =0,1,2 FOR H,M,S
E049                    0550 ; RETURN WITH <H>=0,<L>=HH,MM,SS
E049                    0560 ;  ALL OTHER REGISTERS PRESERVED
```

```
E049 F5              0570 GCLOK  PUSH  PSW
E04A D5              0580       PUSH  D
E04B CD 06 E0        0590       CALL  CLOCK
E04E 21 38 CC        0600       LXI   H,CRTIM
E051 7D              0610       MOV   A,L
E052 83              0620       ADD   E
E053 83              0630       ADD   E
E054 83              0640       ADD   E
E055 6F              0650       MOV   L,A
E056 7E              0660       MOV   A,M
E057 E6 07           0670       ANI   7
E059 07              0680       RLC                ;=*2
E05A 57              0690       MOV   D,A
E05B 07              0700       RLC                ;=*4
E05C 07              0710       RLC                ;=*8
E05D 82              0720       ADD   D            ;=*10
E05E 57              0730       MOV   D,A
E05F 2C              0740       INR   L
E060 7E              0750       MOV   A,M
E061 E6 0F           0760       ANI   15
E063 82              0770       ADD   D
E064 6F              0780       MOV   L,A
E065 26 00           0790       MVI   H,0
E067 D1              0800       POP   D
E068 F1              0810       POP   PSW
E069 C9              0820       RET
E06A                 0830 ; PCLOK RETURNS TOTALIZED MINUTES OR SECONDS:
E06A                 0840 ; CALL WITH <E>=0, <HL> RETURNS (60*HH) + MM
E06A                 0850 ; CALL WITH <E>=1, <HL> RETURNS (60*MM) + SS
E06A                 0860 ; ALL OTHER REGISTERS PRESERVED
E06A                 0870 ; CLOCK SETTING IN VIDEO RAM SUPRESSED AFTER
E06A                 0880 ; FIRST READING OF TIME
E06A F5              0890 PCLOK  PUSH  PSW
E06B D5              0900       PUSH  D
E06C 7B              0910       MOV   A,E
E06D F5              0920       PUSH  PSW
E06E CD 49 E0        0930       CALL  GCLOK        ;<HL> = HH OR MM
E071 EB              0940       XCHG               ;<DE> = HH OR MM
E072 3E 0F           0950       MVI   A,15
E074 21 00 00        0960       LXI   H,0
E077 19              0970 MUL15  DAD   D
E078 3D              0980       DCR   A
E079 C2 77 E0        0990       JNZ   MUL15        ;=*15
E07C 29              1000       DAD   H            ;*30
E07D 29              1010       DAD   H            ;*60
E07E F1              1030       POP   PSW
E07F E5              1040       PUSH  H            ;SAVE *60
E080 3C              1050       INR   A
E081 5F              1060       MOV   E,A
E082 3A 0B 00        1070       LDA   VDMCW        ;SUPRESS THE
E085 F5              1080       PUSH  PSW          ;CLOCK UPDATE!
E086 F6 08           1090       ORI   TIMEB
E088 32 0B 00        1100       STA   VDMCW
E08B CD 49 E0        1110       CALL  GCLOK        ;<HL> = MM OR SS
E08E F1              1120       POP   PSW          ;RESTORE CLOCK
E08F 32 0B 00        1130       STA   VDMCW        ;TO PREVIOUS STATE
E092 D1              1140       POP   D            ;NOW ADD THE *60 VALUE
E093 19              1150       DAD   D            ;<HL> = MMM OR SSS
E094 D1              1160       POP   D
E095 F1              1170       POP   PSW
```

```
E096 C9                 1180        RET
E097                    1190        END

SYMBOL TABLE

CLOCK E006   CLOK1 E010   CPORT 0027   CRTIM CC38   CTABL E043
GCLOK E049   GETIM E000   GPACK E003   MASK  00C0   MUL15 E077
PCLOK E06A   RETRN E01C   SECND CC3F   TIMEB 0008   UPDAT E01E
UTIM1 E038   UTIM2 E035   UTIM3 E032   VDMCW 000B
```

INITIALIZATION OF PARALLEL PORT ON STATION CONTROLLER BOARD (ACB):

```
        IF      ACB     ;ANTENNA CONTROL BOARD
ACBCP   EQU     20H     ;CONTROL PORT
        MVI     A,98H   ;8255 -> A = IN, B&C = OUT
        OUT     ACBCP   ;PROGRAM 8255 MODE
        ENDIF
```

(DE G3ZCZ)

THE FOLLOWING PAGE CONTAINS WIHDK's READ ROUTINE FOR ACB's DIGITAL VOLTMETER. THE FIRST VERSION OF THIS BOARD REQUIRED SOFTWARE SCANNING OF THE BIT-PARALLEL, CHARACTER-SERIAL OUTPUT OF THE MC14433 A/D CONVERTER. THE CURRENT (SECOND) REVISION OF THE BOARD HAS A 4-CHARACTER ON-BOARD RAM (7489) BUFFER TO SAVE THE A/D CONVERTER OUTPUT WHICH SIMPLIFIES SOFTWARE. ASSEMBLY LANGUAGE SOFTWARE DRIVER FOR THE REV. 2 BOARD SHOULD BE AVAILABLE SOON.

H-9

```
                                .PABS
                        ;
                        ;
                        ;THIS ROUTINE READS MC14433P A/D CONVERTER AND STORES
                        ;BCD DIGITS FOR READING BY CALLING PROGRAM. PRIOR TO
                        ;USE 8255 PPI MUST BE SET TO CONTROL WORD 98H AND THE
                        ;MUX ADDRESS MUST BE OUTPUT TO SELECT DESIRED CHANNEL.
                        ;
                        ;
6E00                            .LOC 6E00H
                        ;
6E00    DB22    EOC:    IN      22H     ;CHECK THE EOC BIT
6E02    E680            ANI     80H     ;AND LOOP UNTIL TRUE
6E04    CA 6E00         JZ      EOC
6E07    DB20    DS1:    IN      20H     ;LOOK AT THE DIGIT STROBES
6E09    E680            ANI     80H     ;AND LOOP UNTIL D1 GOES TRUE
6E0B    CA 6E07         JZ      DS1
6E0E    DB20            IN      20H     ;THEN READ THE MSD
6E10    47              MOV     B,A     ;AND STORE IN B REGISTER
6E11    DB20    DS2:    IN      20H     ;LOOK AT STROBES AGAIN
6E13    E640            ANI     40H     ;AND LOOP UNTIL D2 GOES TRUE
6E15    CA 6E11         JZ      DS2
6E18    DB20            IN      20H     ;THEN READ 2ND MSD
6E1A    4F              MOV     C,A     ;AND STORE IN C REGISTER
6E1B    DB20    DS3:    IN      20H     ;LOOK AT DIGIT STROBES AGAIN
6E1D    E620            ANI     20H     ;AND LOOP UNTIL D3 GOES TRUE
6E1F    CA 6E1B         JZ      DS3
6E22    DB20            IN      20H     ;THEN READ 3RD MSD
6E24    57              MOV     D,A     ;AND STORE IN D REGISTER
6E25    DB20    DS4:    IN      20H     ;LAST LOOK AT DIGIT STROBES
6E27    E610            ANI     10H     ;AND LOOP UNTIL D4 GOES TRUE
6E29    CA 6E25         JZ      DS4
6E2C    DB20            IN      20H     ;THEN READ LSD
6E2E    32 6E42         STA     D4      ;STORE LSD IN LOCATION D4
6E31    7A              MOV     A,D     ;GET BACK 3RD MSD
6E32    32 6E41         STA     D3      ;AND STORE IN LOCATION D3
6E35    79              MOV     A,C     ;GET BACK 2ND MSD
6E36    32 6E40         STA     D2      ;AND STORE IN LOCATION D2
6E39    78              MOV     A,B     ;GET BACK MSD
6E3A    32 6E3F         STA     D1      ;AND STORE IN LOCATION D1
                        ;
6E3D    F9              SPHL            ;RESTORE CALLING STACK
6E3E    C9              RET             ;AND RETURN
                        ;
                        ;
6E3F                    D1=.    ;USE THESE LOCATIONS FOR DIGIT STORAGE
6E40                    D2=.+1
6E41                    D3=.+2
6E42                    D4=.+3
                        ;
                        ;
                                .END
```

Figure ~~A-6.~~ Assembly listing for A/D converter "reader". (DE W1HDX)

2708 EPROM ZAPPER IN N* BASIC USING AMSAT N* → AMS80 INTERFACE UTILITIES.

WORKS WITH CROMEMCO BYTESAVER.

(H-10)

LIST#1

```
10!CHR$(12),"ZAP = 2708 EPROM ZAPPER"
20 Z0=FNA("29CD") \ Z1=FNA("77") \ REM-TO MODIFY VERIFY CODE
30 DIM L(5) \ GOTO 240
40REM--2708 EPROM ZAPPER
50REM--ROUTINE TO CONVERT FROM ASCII TO HEX
60DEF FNH(V$,I)
70V=ASC(V$(I,I))- 48
80IFV<0THEN100 ELSE IFV>9ANDV<17THEN100 ELSE IF V>22THEN 100
90IF V<10 THEN RETURN V ELSE RETURN V-7
100!CHR$(7),"ERROR IN HEX INPUT ",V$(I,I)\STOP
110FNEND
120REM--ROUTINE TO PARSE HEX ADDRESS, RETURNING INTEGER
130DEFFNA(V$)                    (ASCII HEX $TRING → DECIMAL NUMBER)
140L=LEN(V$)\IFL>0ANDL<5 THEN150 ELSE!CHR$(7),"LENGTH ERROR"\STOP
150V=0\FORI=1TOL\V=16*V+FNH(V$,I)\NEXT\RETURNINT(V+.1)
160FNEND
170REM--ROUTINE TO CONVERT ADDRESS TO HEX,LENGTH L
180DEFFNH$(I,L)          (DECIMAL NUMBER of Length L [1≤L≤4]
190I9=I\H$=""                  TO HEX $TRING)
200FORJ9=1TOL\H9=16↑(L-J9)\K9=INT(I9/H9)
210I9=I9-K9*H9
220H$=H$+CHR$(48 + 7*(K9>9) +K9) \ NEXT \ RETURNH$
230FNEND
240INPUT"SOURCE START ADDRESS =",S$
250S0=FNA(S$)
260INPUT"SOURCE END ADDRESS =",E$
270S1=FNA(E$)
280!"2708 ZAPPER--SOURCE =",S$,"=",S0,"  TO=",E$,"=",S1
290 N2=(S1-S0+1)/256 \ IF N2=INT(N2) THEN 330
300 !"REQUESTED NON-INTEGRAL ## 256 BYTE BLOCKS = ",N2
310 !"THIS ZAP VERSION SUPPORTS ONLY BLOCK TRANSFERS"
320 !"TRY AGAIN----",CHR$(7), \ GOTO 240
330INPUT"DESTINATION START ADDRESS=",S$
340D0=FNA(S$)\D1=S1-S0+D0\E$=FNH$(D1,4)
350!"2708 ZAPPER--DEST'N ="",S$,"="",D0,"  TO=",E$,"="",D1
360!"TOTAL TRANSFER WILL BE",N2," 256 BYTE BLOCKS"
370 INPUT"## 256 BYTE TRANSFERS PER STEP = ",K1
380 IF N2/K1 = INT(N2/K1) THEN 410
390 !CHR$(7),"NON-INTEGRAL NUMBER OF STEPS RESULTS. TRY AGAIN"
400 GOTO 370
410INPUT"## OF LOOPS (100?) ?",N
420FORN1=1TON \ OUT255,255-N1
430!\!"LOOP #",N1\K=-1
440 FOR S=S0 TO S1 STEP 256*K1 \ K=K+1 \ D=D0+K*256*K1
450 FILL Z0,Z1 \ REM--NOW WRITE IT
460 FILL 10666,K1 \ Z=CALL(10647,D) \ Z=CALL(10682,S)   ← LINKAGES TO N* INTERFACE UTILITIES!
470 !"STEP #",K,"  FROM=",FNH$(S,4),"-",FNH$(S+K1*256-1,4),
480 !"   TO=",FNH$(D,4),"-",FNH$(D+256*K1-1,4),
490 FILL Z0,0 \ Z=CALL(10682,S) \ FILL Z0,Z1 \ REM-READBACK
500 IF Z THEN L(K)=N1 \ !"  ERR=",Z,"  LAST=",L(K)
510 NEXTS \ NEXTN1
520!\!"END" \ END
READY
```

USEFUL HEX PARSING UTILITIES

THIS ROUTINE IS AS FAST AS ANY COMMERCIAL 2708 ZAPPER HARDWARE I HAVE SEEN! 4-2708's ZAP IN < 5 MINUTES.

(DE W3IWI)

IF YOU WANT A UTILITY TO SEARCH MEMORY FOR 16 bit #'s BETWEEN 2 LIMITS AND REPORT ALL OCCURRENCES, HERE IT IS!

```
 LOAD FINDIT,2
READY
LIST#1

10 REM--FINDIT PROGRAM FINDS ALL OCCURENCES OF AN ADDRESS RANGE
20 REM  IN A BLOCK OF MEMORY.      T.A.CLARK    MAY 1979
30 INPUT"PRINTER UNIT # =",P
40!#P,CHR$(12),"FINDIT -- FIND MEMORY REFFERENTS"
50 GOTO 250
60REM--ROUTINE TO CONVERT FROM ASCII TO HEX
70DEF FNH(V$,I)
80V=ASC(V$(I,I))- 48
90IFV<0THEN110 ELSE IFV>9ANDV<17THEN110 ELSE IF V>22THEN 110
100IF V<10 THEN RETURN V ELSE RETURN V-7
110!CHR$(7),"ERROR IN HEX INPUT ",V$(I,I)\STOP
120FNEND
130REM--ROUTINE TO PARSE HEX ADDRESS, RETURNING INTEGER
140DEFFNA(V$)
150L=LEN(V$)\IFL>0ANDL<5 THEN160 ELSE!CHR$(7),"LENGTH ERROR"\STOP
160V=0\FORI=1TOL\V=16*V+FNH(V$,I)\NEXT\RETURNINT(V+.1)
170FNEND
180REM--ROUTINE TO CONVERT ADDRESS TO HEX,LENGTH L
190DEFFNH$(I,L)
200I9=I\H$=""
210FORJ9=1TOL\H9=16↑(L-J9)\K9=INT(I9/H9)
220I9=I9-K9*H9
230H$=H$+CHR$(48 + 7*(K9>9) +K9) \ NEXT \ RETURNH$
240FNEND
250INPUT"SOURCE START ADDRESS = ",S$
260S0=FNA(S$)
270INPUT"SOURCE END ADDRESS = ",E$
280S1=FNA(E$)
290 INPUT"MIN DATA VALUE = ",M0$
300 M0=FNA(M0$) \ X0=INT(M0/256)
310 INPUT"MAX DATA VALUE = ",M9$
320 M9=FNA(M9$) \ X9=INT(M9/256) \!
330 FOR I=S0 TO S1 \ X=EXAM(I) \ IF X<X0 THEN 360 ELSE IF X>X9 THEN 360
340 M=256*X+EXAM(I-1) \ IF M<M0 THEN 360 ELSE IF M>M9 THEN 360
350 ! "ADDR=",FNH$(I-1,4),"    DATA=",FNH$(M,4)
360 NEXT
370!\!"END" \ END
READY
```

M0$, M9$ ARE 16 bit (4 Hex character) NUMBERS!

← SEARCHES MSBYTE THEN
← SEARCHES LS BYTE

LIST#7

```
10 REM *  PROGRAM FOR PRODUCING AZ-EL POINTING DATA                *
20 REM *  FOR THE AMSAT-OSCAR-7 AND -8 SATELLITES.                 *
30 REM *  THIS VERSION FOR NORTH-STAR DISC BASIC.                  *
40 REM *  OTHER VERSIONS ALSO AVAILABLE ON REQUEST                 *
50 REM *             MAY, 1978                                     *
60 REM *  BY    DR. T. A. CLARK    W3IWI/WA3LND                    *
70 REM *        % AMSAT HEADQUARTERS                              *
80 REM *        P.O. BOX 27                                       *
90 REM *        WASHINGTON, D.C. 20044                            *
100REM *        (301) 286-3113 AT HOME                            *
110REM *
120 P5=0 \ REM-PRINTER
130 C5=0 \ REM CRT
140 DIM I$(30) \P1=3.141592654\P2=2*P1\P8=P1/180 \ C=2.998E5
150 GOTO 510
160 REM -- FNZ$ FILLS IN LEADING ZEROES IN STRING OF LEN=2
170 DEF FNZ$(V)
180 V1=INT(V/10) \ V2=INT(V-10*V1+.5)
190 RETURN CHR$(48+V1)+CHR$(48+V2)
200 FNEND
210 REM -- FNT CONVERTS FROM DD.DDD TO HH,MM,SS
220 DEF FNT(D9)
230 T7=INT(D9) \ T9=D9-T7 \ H9=INT(24*T9)
240 M9=INT(1440*(T9-H9/24))
250 S9=INT((T9-H9/24-M9/1440)*86400+.5)
260 IF S9<60 THEN 270 ELSE S9=S9-60 \ M9=M9+1 \ GOTO 260
270 IF M9<60 THEN 280 ELSE M9=M9-60 \ H9=H9+1 \ GOTO 270
280 IF H9<24 THEN 290 ELSE H9=H9-24 \ T7=T7+1 \ GOTO 280
290 RETURN T7
300 FNEND
310 REM -- GET SATELLITE COORD,AZ,EL,RANGE AT T=DD.DDD
320 DEF FNS(T9)
330 A0=(T9-T0)+W0 \ A0=(A0-INT(A0))*P2
340 N9=(T9-T0)/P0 \ A1=(N9-INT(N9))*P2
350 S0=SIN(A0) \ C0=COS(A0) \ S1=SIN(A1) \ C1=COS(A1)
360 X=R3*(+C0*C1+S0*S1*C2) \ Y=R3*(-S0*C1+C0*S1*C2)
370 Z=R3*S1*S2 \ S8=(X-X0)↑2+(Y-Y0)↑2+(Z-Z0)↑2 \ S7=SQRT(S8)
380 S3=(R4-S8)/(2*R0*S7) \ E=-99 \F7=(S7-L7)/(T5*C) \ L7=S7
390 IF S3<E3 THEN RETURN E ELSE E=ATN(S3/SQRT(1-S3↑2))/P8
400 A1=R1*(Y*X0-X*Y0) \ A2=R7*Z - Z0*(X*X0+Y*Y0+Z*Z0)
410 A=ATN(A1/A2)/P8 \ IF A2<0 THEN A=A+180
420 IF (A1<0 AND A2>0) THEN A=A+360
430 A=INT(A+.5) \ S7=INT(S7+.5) \ RETURN INT(E+.5)
440 FNEND
450 REM -- HEADER PRINT ROUTINE
460 T6=T0+(N-N0)*P0 \ W6=T6-T0+W0 \ W6=(W6-INT(W6))*360
470 D7=FNT(T6)+D \ T$="/"+FNZ$(H9)+":"+FNZ$(M9)+":"+FNZ$(S9)
480 !"ORBIT# ",%5I,N," AT ",D7,T$," UTC  AT W.LONG=",%6F2,W6
490 !"    UTC         AZ        EL     RANGE       "+M$ \ RETURN
500 REM ---- PROGRAM BEGINS
510 !$C5,CHR$(12),"AMSAT-OSCAR AZ-EL PROGRAM"
520 INPUT"WHICH SATELLITE ",S
530 IF S<>7 THEN 540 ELSE RESTORE 980 \ GOTO 560
540 IF S<>8 THEN 550 ELSE RESTORE 1060 \ GOTO 560
550 IF S=0 THEN STOP ELSE !"INVALID ENTRY -- ", \ GOTO 520
560 READ I$,N0,D,H9,M9,S9,W,P,I,H
570 INPUT"DESIRED MODE = ",M$
580 READ T$,F8,F9\IF T$=M$THEN590 ELSEIF T$="**"THEN610 ELSE580
```

COPY ON NORTH-STAR DISC AVAILABLE

SELECTS I/O UNITS

π 2π π/180 VEL OF LIGHT

SETS UP "PRETTY" TIMES FOR PRINTOUT

TIMES ARE KEPT INTERNALLY IN DAYS & FRACTIONS, BUT PEOPLE LIKE HOURS+MINUTES.

TRIG TO GET THE SATELLITE'S X,Y,Z COORD'S AND RANGE

THESE ALGORITHMS ARE VALID ONLY FOR A CIRCULAR SATELLITE ORBIT THAT IS SUN-SYNCHRONOUS LIKE AO-6,7,8

RESOLVES QUADRANT AMBIGUITY

RANGE DIFFERENCES BETWEEN STEPS ARE CONVERTED TO DOPPLER OR VELOCITY FOR PRINTOUT

IS SATELLITE ABOVE MINIMUM ELEVATION? IF SO, COMPUTE AZ & EL

```
590 IF F8=1 THEN M$="BEACON" ELSE M$="DOPPLER" \ F8=F8*F9
600 M$=T$(1,1)+"-"+M$ \ GOTO 620
610 M$="  VELOCITY" \ F8=0 \ F9=C
620 RESTORE 1140 \ READ E3,T5,S$,W1,L1
630 P0=P/1440 \ W0=W/360 \ T0=H9/24+M9/1440+S9/86400
640 S2=SIN(I*P8) \ E3=SIN(E3*P8) \ C2=COS(I*P8) \ R1=6378.14
650 R3=R1+H \ R7=R1↑2 \ R5=R3↑2
660 S0=SIN(L1*P8)\C0=COS(L1*P8)\S1=SIN(W1*P8)\C1=COS(W1*P8)
670 F=1-1/298.22 \ R=R1/SQRT(C0*C0+F*S0*S0)
680 X0=R*C0*C1 \ Y0=-R*C0*S1 \ Z0=R*S0*F*F
690 R0=SQRT(X0*X0+Y0*Y0+Z0*Z0) \ R4=R5-R0*R0
700 INPUT"START,END ORBIT # = ",N1,N2\IF N2<=0 THEN N2=N1
710 IF N2<N1 THEN N2=N1+N2-1 \ I$=I$+" ORBITS "
720 !#P5,CHR$(12),"AZ-EL POINTING FOR ",I$,N1," TO ",N2
730 I$="FOR "+S$+" AT W.LONG="
740 !I$,W1," & LAT=",L1,"  MODE-"+T$+": F=",F9
750 !"-------------------------------------------------------"
760 T1=T0+(N1-N0)*P0 \ T1=INT(T1*1440)/1440
770 T2=T1+(N2-N1+1)*P0
780 T3=T5/1440 \ T5=T5*60 \ L7=9E9 \ N7=-99
790 FOR T=T1 TO T2 STEP T3 \ E = FNS(T) \ IF E<-90 THEN 830
800 N=INT(N0+N9) \ IF N<>N7 THEN GOSUB 460
810 N7=N \ D7=FNT(T) \ T$="   "+FNZ$(H9)+":"+FNZ$(M9)
820 !T$,%9I,A,E,S7,%14F2,F8-F7*F9
830 NEXT \ !TAB(20),"END"
840REM
850REM *  ORBITAL DATA THAT FOLLOWS SHOULD APPLY TO  *
860REM *  A DATE WITHIN A MONTH OR TWO OF THE DATE   *
870REM *  OF INTEREST TO MINIMIZE ERROR PROPAGATION. *
880REM *  ORBITAL DATA IS AVAILABLE VIA AMSAT FROM   *
890REM *        SKIP REYMANN, W6PAJ                  *
900REM *        P.O. BOX 374                         *
910REM *        SAN DIMAS, CALIF 91773               *
920REM *  FOR $5.00 ($3.00 FOR AMSAT MEMBERS AND     *
930REM *  FREE TO AMSAT LIFE MEMBERS). MEMBERSHIP    *
940REM *  INFO CAN BE OBTAINED FROM AMSAT AT THE     *
950REM *  ADDRESS AT THE TOP. PLEASE JOIN 'CUZ THE   *
960REM *  AMATEUR SATELLITE PROGRAM NEEDS YOUR HELP! *
970REM
980 DATA          "AMSAT-OSCAR-7"
990 REM  ORBIT, DATE,EQCR TIME & LONG,  PERIOD   , INCL,HEIGHT
1000 REM   ##  ,YYDDD,HH,MM,SS.S, DEG.,   MINS.   , DEG.,  KM
1010 DATA 21377,79200,01,20,10,84.2,114.944814 ,101.7,1449.6
1020 REM FORMAT="MODE",DOP=0/FREQ=1,EQUIV FREQ,KHZ
1030 DATA "A",0,175450.00,"AB",1,29502.00
1040 DATA "B",0,-286250.00,"BB",1,145972.00
1050 DATA "D",1,435100.00,"**",1,1 \ REM ** TERMINATES
1060 DATA          "AMSAT-OSCAR-8"
1070 REM  ORBIT, DATE,EQCR TIME & LONG,  PERIOD   , INCL,HEIGHT
1080 REM   ##  ,YYDDD,HH,MM,SS.S, DEG.,   MINS.   , DEG.,  KM
1090 DATA 07035,79204,00,44,05,56.5,103.22063 ,98.99,905
1100 DATA "A",0,175450.00,"AB",1,29402.00
1110 DATA "J",0,+289100.00,"JB",1,435100.00
1120 DATA "**",1,1 \ REM ** TERMINATES
1130 REM - - - STATION INFO - - -
1140 DATA -3,2 \ REM-MIN EL,TIME INTERVAL
1150 DATA "W3IWI",76.938,39.201,0 \ REM-CALL,LONG,LAT, DUMMY TO MARK END OF TABLE
1160 END
READY
```

PREPARE FOR DOPPLER OR VELOCITY CALCULATIONS.

PRINTS SATELLITE VELOCITY IF INVALID MODE WAS REQUESTED

SET UP SATELLITE PARAMETERS

OBSERVER'S COORDINATES ON OBLATE SHEROIDAL EARTH

CONVERT ORBIT # TO TIME RANGE, TIME IN DAYS

STEP AROUND ORBIT IN STEPS OF (15) MINUTES, CHECKING IF SATELLITE IS VISIBLE.

SAMPLE RUNS

```
AMSAT-OSCAR AZ-EL PROGRAM
WHICH SATELLITE 8
DESIRED MODE = J
START,END ORBIT # = 7508,7510
```

ASKED FOR SPECIFIC ORBITS

```
AZ-EL POINTING FOR AMSAT-OSCAR-8 ORBITS  7508 TO  7510
FOR W3IWI AT W.LONG= 76.938 & LAT= 39.201  MODE-J: F= -371.5
-----------------------------------------------------------
ORBIT#  7508 AT 79237/22:27:26 UTC  AT W.LONG= 22.34
    UTC        AZ        EL     RANGE      J-DOPPLER
  22:41        62        -2      3760           2.43
  22:43        48        -1      3612           1.19
  22:45        34        -1      3641           -.23
  22:47        21        -3      3843          -1.62
ORBIT#  7509 AT 79238/00:10:40 UTC  AT W.LONG= 48.14
    UTC        AZ        EL     RANGE      J-DOPPLER
  00:17       123         3      3248           5.59
  00:19       113        10      2612           5.11
  00:21        96        18      2107           4.66
  00:23        69        23      1854           2.03
  00:25        40        20      1958           -.84
  00:27        20        13      2372          -3.33
  00:29         7         6      2965          -4.76
  00:31       360        -1      3644          -5.46
ORBIT#  7510 AT 79238/01:53:53 UTC  AT W.LONG= 73.95
    UTC        AZ        EL     RANGE      J-DOPPLER
  01:57       179         1      3444           6.18
  01:59       183         9      2685           6.10
  02:01       191        20      1965           5.78
  02:03       210        38      1369           4.79
  02:05       266        52      1126           1.95
  02:07       315        35      1432          -2.46
  02:09       331        19      2032          -4.98
  02:11       337         8      2776          -5.82
  02:13       341         0      3534          -6.09
                      END

READY
RUN
```

OUTPUT - INPUT FREQ

DOPPLER IS COMPUTED FROM FINITE DIFFERENCES OF RANGE. TIME IS OFFSET BY ½ STEP, E.G. 2.03 KHZ REALLY APPLIES TO 00:22 UT

```
AMSAT-OSCAR AZ-EL PROGRAM
WHICH SATELLITE 8
DESIRED MODE = JB
START,END ORBIT # = 7412,1
```

ASKED FOR A # OF ORBITS (=1)

```
AZ-EL POINTING FOR AMSAT-OSCAR-8 ORBITS  7412 TO  7412
FOR W3IWI AT W.LONG= 76.938 & LAT= 39.201  MODE-JB: F= 435107
-----------------------------------------------------------
ORBIT#  7412 AT 79231/01:18:16 UTC  AT W.LONG= 65.04
    UTC        AZ        EL     RANGE      J-BEACON
  01:22       161         2      3341      435107.49
  01:24       160        11      2561      435107.43
  01:26       158        24      1812      435107.06
  01:28       151        48      1175      435107.71
  01:30        32        77       938      435102.86
  01:32       354        39      1338      435095.16
  01:34       350        19      2023      435091.71
  01:36       349         8      2785      435090.79
  01:38       348         0      3567      435090.55
                      END

READY
```

NOMINAL MODE-J BEACON FREQ

H-15

```
AMSAT-OSCAR AZ-EL PROGRAM
WHICH SATELLITE ?
DESIRED MODE = A
START,END ORBIT # = 21927,2

AZ-EL POINTING FOR AMSAT-OSCAR-7 ORBITS  21927 TO  21928
FOR W3IWI AT W.LONG= 76.738 & LAT= 39.2N1  MODE-A: F= 175450
-------------------------------------------------------
ORBIT# 21927 AT 79243/22:59:49 UTC  AT W.LONG= 49.11
    UTC        AZ        EL     RANGE      A-DOPPLER
   23:03       136       -1      4838         3.32
   23:05       132        5      4518         3.27
   23:07       127       13      3378         3.15
   23:09       118       22      2778         2.89
   23:11       104       32      2295         2.36
   23:13        78       41      2078         1.41
   23:15        45       41      2093          .41
   23:17        20       32      2296        -1.39
   23:19         6       22      2774        -2.36
   23:21       358       13      3386        -2.89
   23:23       352        5      4014        -3.18
   23:25       349       -1      4689        -3.29
ORBIT# 21928 AT 79244/00:54:45 UTC  AT W.LONG= 77.85
    UTC        AZ        EL     RANGE      A-DOPPLER
   00:55       182       -3      4851         3.30
   00:57       185        4      4180         3.27
   00:59       190       11      3336         3.17
   01:01       198       19      2923         2.98
   01:03       211       29      2403         2.54
   01:05       233       40      2044         1.73
   01:07       266       43      1943          .49
   01:09       296       37      2136         -.74
   01:11       314       26      2555        -2.44
   01:13       325       16      3106        -2.88
   01:15       331        8      3724        -3.02
   01:17       336        2      4376        -3.18
                     END
READY
RUN
```

SINCE DOPPLER APPLIES ½ STEP EARLIER, NOTICE THAT TCA WAS 23:14

A 2900 1900

NORTH-STAR TO AMS-80 INTERFACES

N*1

SINGLE DENSITY

```
2900                    0010 * NORTH-STAR RELEASE 4 I/O AND INTERRUPT DRIVERS
2900                    0020 * T.A.CLARK,W3IWI   25 AUG 78
2900                    0030         ORG   2000H
2000 59                 0040 TRKID   DB    59H           ;CURRENT TRK NUMBER
2001 59                 0050         DB    59H           ;BY DRIVE #, 59H
2002 59                 0060         DB    59H           ;FOR UNDEFINED.
2003                    0070         DS    10            ;OTHER N* STORAGE
2000 C3 3C 29           0080 COUT    JMP   XOUT
2010 C3 5C 29           0090 CIN     JMP   XIN
2013 C3 00 29           0100 TINIT   JMP   START
2016 C3 6C 29           0110 CONTC   JMP   XCNTC
2019                    0120 HDERR   DS    3             ;DISK ERROR HANDLER
201C                    0130 DLOOK   DS    3             ;DIRECTORY LOOK-UP
201F                    0140 DWRIT   DS    3             ;DIRECTORY RE-WRITER
2022 C3 89 29           0150 DXCOM   JMP   XCOM          ;PATCHED DISK I/O ENTRY
2025                    0160 LIST    DS    3             ;DIRECTORY LISTER
2028                    0170 ENTRY   DS    3             ;RESTART ENTRY POINT
2028 01                 0180 RWCHK   DB    1             ;READ-AFTER WRITE=ON
202C                    0190 * INTERRUPT DRIVER PATCHES IN N* DOS:
202C                    0200         ORG   2455H
2455 C3 89 29           0210 DCOM    JMP   XCOM
2458                    0220 * THE FOLLOWING PERSONALIZES THE
2458                    0230 * #LINES/PAGE TO YOUR REQUIREMENTS:
2458                    0240         ORG   28F8H
28F8 10                 0250 LINES   DB    16
28F9                    0260 * AMS80 ENTRY POINTS:
28F9                    0270 AMS80   EQU   0F000H
28F9                    0280 CONSI   EQU   AMS80+3
28F9                    0290 RDRIN   EQU   AMS80+6
28F9                    0300 CONSO   EQU   AMS80+9
28F9                    0310 CSTS    EQU   AMS80+12H
28F9                    0320 IOCHK   EQU   AMS80+15H
28F9                    0330 IOSET   EQU   AMS80+18H
28F9                    0340 MSG     EQU   AMS80+3CH
28F9                    0350 CONSB   EQU   AMS80+42H
28F9                    0360 PNCHB   EQU   AMS80+51H
28F9                    0370 LISTB   EQU   AMS80+5DH
28F9                    0380 IOCW    EQU   3
28F9                    0390 VDMCW   EQU   0BH
28F9                    0400 VDM     EQU   AMS80+0D00H
28F9                    0410 *
28F9                    0420 * INITIALIZATION ROUTINE = 10496D
28F9                    0430 * TERMINAL INITIALIZATION ROUTINE
28F9                    0440         ORG   2900H
2900 21 19 29           0450 START   LXI   H,IOTBL       ;I/O SETUP TABLE
2903 7E                 0460         MOV   A,M
2904 32 03 00           0470         STA   IOCW
2907 23                 0480         INX   H
2908 7E                 0490         MOV   A,M
2909 32 0B 00           0500         STA   VDMCW
290C 23                 0510         INX   H
290D CD 3C F0           0520         CALL  MSG
2910 21 59 59           0530         LXI   H,5959H
2913 22 00 20           0540         SHLD  TRKID         ;RESET TRACK POINTERS
2916 C3 29 20           0550         JMP   ENTRY
2919                    0560 * IOTBL @ 10521D
2919 00                 0570 IOTBL   DB    00H           ;DEFAULT IOCW ] SET TO MEET YOUR NEEDS
```

```
291A 80                     0580          DB    80H           ;DEFAULT VIDCW
2919 0C                     0590 MSGXX    DB    0CH           ;CLEAR SCREEN
291C 4E 2A 20 44 4F         0600          DB    'N* DOS REV 4 / AMS80 REV 5.8'
     53 20 52 45 56
     20 34 20 2F 20
     41 4D 53 38 30
     20 52 45 56 20
     35 2E 38
2938 0A                     0610          DB    0AH           ;LINE FEED
2939 0D                     0620          DB    0DH           ;CAR RET
293A 07                     0630          DB    7             ;RING BELL
293B FF                     0640          DB    0FFH          ;END OF MSG
293C                        0650 * I/O DRIVERS
293C E6 07                  0660 XOUT     ANI   7
293E FE 05                  0670          CPI   5
2940 FA 55 29               0680          JM    OUT04
2943 FE 06                  0690          CPI   6
2945 78                     0700          MOV   A,B
2946 FA 51 F0               0710 OUT5     JM    PNCHB
2949 CA 5D F0               0720 OUT6     JZ    LISTB
294C C5                     0730 OUT7     PUSH  B
294D 48                     0740          MOV   C,B
294E CD 00 FD               0750          CALL  VDM
2951 C1                     0760          POP   B
2952 C3 5D F0               0770          JMP   LISTB
2955 CD 78 29               0780 OUT04    CALL  IOASN         ;DEVICES 0-4
2958 78                     0790          MOV   A,B
2959 C3 42 F0               0800          JMP   CONSB
295C E6 07                  0810 XIN      ANI   7
295E FE 05                  0820          CPI   5
2960 CA 06 F0               0830 IN5      JZ    RDRIN
2963 F2 03 F0               0840 IN67     JP    CONSI
2966 CD 78 29               0850 IN04     CALL  IOASN         ;DEVICES 0-4
2969 C3 03 F0               0860          JMP   CONSI
296C                        0870 *
296C                        0880 * CNTL-C DETECTOR ALWAYS TESTS CURRENT CONSOLE!
296C CD 12 F0               0890 XCNTC    CALL  CSTS
296F FE FF                  0900          CPI   0FFH          ;WAS A KEY STRUCK?
2971 C0                     0910          RNZ                 ;NOPE
2972 CD 03 F0               0920          CALL  CONSI         ;GET THE CHARACTER
2975 FE 03                  0930          CPI   3             ;=CNTL-C?
2977 C9                     0940          RET
2978                        0950 * IOASN HANDLES I/O DEVICES (0-4) WITH:
2978                        0960 *  DEVICE 0 = CURRENT CONSOLE
2978                        0970 *  DEVICES (1-4) = AMS80 CONSOLE DEVICES (0-3)
2978                        0980 *   WHERE AFTER A CALL TO DEVICE (1-4), THE
2978                        0990 *   CURRENT CONSOLE SWITCH IN IOCW IS MODIFIED
2978                        1000 *   UNTIL A SUBSEQUENT CALL WITH A DIFFERENT
2978                        1010 *   DEVICE ID.
2978 D6 01                  1020 IOASN    SUI   1             ;SEPARATE 0 FROM (1-4)
297A F8                     1030          RM                  ;IF DEVICE=0
297B C5                     1040          PUSH  B
297C 4F                     1050          MOV   C,A
297D CD 15 F0               1060          CALL  IOCHK
2980 E6 FC                  1070          ANI   0FCH
2982 B1                     1080          ORA   C
2983 4F                     1090          MOV   C,A
2984 CD 18 F0               1100          CALL  IOSET
2987 C1                     1110          POP   B
2988 C9                     1120          RET
```

N*2

SET TO MEET YOUR NEEDS.

REPLACE WITH 0FFH IF YOU WANT TO SUPRESS THE BELL.

N*3

```
2989                    1130 *
2989                    1140 * PATCHES TO SUPPORT INTERRUPTS IN N* DOS:
2989                    1150 * THE GENERAL PHILOSOPHY IS TO DISABLE ALL
2989                    1160 * INTERRUPTS WHILE ANY DISK I/O IS OCCURRING.
2989 F3                 1170 XCOM   DI
298A E5                 1180        PUSH  H
298B 21 95 29           1190        LXI   H,RETRN       ;RETURN ADDR
298E E3                 1200        XTHL
298F D5                 1210        PUSH  D
2990 C5                 1220        PUSH  B
2991 F5                 1230        PUSH  PSW
2992                    1240 * THE FOLLOWING INSTRUCTION IS REALLY A CALL WITH
2992                    1250 * THE RETURN ADDRESS TO RETRN 'HIDDEN' ON THE
2992                    1260 * STACK BELOW THE PUSHES:
2992 C3 58 24           1270        JMP   DCOM+3
2995                    1280 * THE FOLLOWING ENTRY POINT MAY BE USED TO ENABLE
2995                    1290 * INTERRUPTS FROM BASIC:
2995 FB                 1300 RETRN  EI
2996 C9                 1310        RET
2997                    1320 *
2997                    1330 * BASIC CALLABLE DIRECT DISK READ/WRITE UTILITY
2997                    1340 *
2997                    1350 * FROM BASIC, DO 3 FILLS:
2997                    1360 *  FILL ABC  ,## 256 BYTE BLOCKS
2997                    1370 *  FILL ABC+1, 1=READ OR 0=WRITE
2997                    1380 *  FILL ABC+2, DRIVE ## (1,2 OR 3)
2997                    1390 * THEN PASS THE DISK ADDRESS WITH A CALL
2997                    1400 *  CALL DSKAD,DISK TRK/SECTOR
2997                    1410 * FINALLY DO THE DIRTY DEED WITH
2997                    1420 *  CALL DSKRW,STARTING RAM ADDRESS
2997                    1430 *
2997                    1440 * WHERE THE DECIMAL EQUIVALENTS FOR ALL THE
2997                    1450 * BASIC ENTRY POINTS AND TABLES ARE:
2997                    1460 *   DSKAD=10647 FOR SAVING DISK/DEST ADDR
2997                    1470 *   DSKRW=10652 FOR DISK/MEMORY MOVES
2997                    1480 *   ABC  =10666,7,8 FOR 3 CONTROL BYTES
2997                    1490 *   MOVE =10682 FOR MEMORY-TO-MEMORY MOVE
2997                    1500 *   BFILL=10717 TO FILL THE BFR
2997                    1510 *   BLOOK=10732 TO GET TRK/SECTOR FOR FILE
2997                    1520 *   RETRN=10645 TO ENABLE INTERRUPTS
2997                    1530 *
2997 EB                 1540 DSKAD  XCHG                ;ADR TO HL
2998 22 AD 29           1550        SHLD  DADDR
299B C9                 1560        RET
299C 21 AA 29           1570 DSKRW  LXI   H,ABC
299F 7E                 1580        MOV   A,M           ;# 256 BYTE BLOCKS
29A0 2C                 1590        INR   L
29A1 46                 1600        MOV   B,M           ;1/0 = READ/WRITE
29A2 2C                 1610        INR   L
29A3 4E                 1620        MOV   C,M           ;DRIVE ##
29A4 2A AD 29           1630        LHLD  DADDR         ;DISK TRK/SECTOR ADDR
29A7 C3 89 29           1640        JMP   XCOM
29AA                    1650 *
29AA                    1660 * UTILITY LINKAGE BUFFER AREA:
29AA                    1670 BFLEN  EQU   16
29AA                    1680 BFR    DS    BFLEN
29BA                    1690 ABC    EQU   BFR
29BA                    1700 DADDR  EQU   BFR+3
29BA                    1710 *
29BA                    1720 * THE FOLLOWING UTILITY IS QUITE
```

```
29BA                    1730 * SIMILAR TO THE DISK/MEMORY UTILITY,
29BA                    1740 * BUT HANDLES MEMORY/MEMORY TRANSFERS:
29BA                    1750 * FIRST SET THE LENGTH WITH
29BA                    1760 *
29BA                    1770 *   FILL ABC,## 256 BYTE BLOCKS
29BA                    1780 * THEN SET THE DESTINATION ADDRESS WITH:
29BA                    1790 *   CALL DSKAD,DESTINATION STARTING ADDR
29BA                    1800 * FINALLY, DO THE MOVE WITH
29BA                    1810 *   CALL MOVE,SOURCE STARTING ADDR
29BA                    1820 *
29BA                    1830 * ON RETURN VIA HL YOU WILL GET THE
29BA                    1840 * THE NUMBER OF ERRORS IN THE MOVE
29BA                    1850 * THIS IS USEFUL FOR PROM ZAPPING USE.
29BA                    1860 *
29BA 3A AA 29           1870 MOVE   LDA   ABC
29BD 47                 1880        MOV   B,A
29BE 0E 00              1890        MVI   C,0           ;BC = ## BYTES
29C0 21 00 00           1900        LXI   H,0           ;FOR
29C3 E5                 1910        PUSH  H             ;ERROR COUNTER
29C4 2A AD 29           1920        LHLD  DADDR
29C7 78                 1930 MLOOP  MOV   A,B
29C8 B1                 1940        ORA   C
29C9 CA DB 29           1950        JZ    MDONE
29CC 1A                 1960        LDAX  D
29CD 77 ?? ØØ           1970        MOV   M,A
29CE BE                 1980        CMP   M
29CF 13                 1990        INX   D
29D0 23                 2000        INX   H
29D1 0B                 2010        DCX   B
29D2 CA C7 29           2020        JZ    MLOOP
29D5 E3                 2030        XTHL
29D6 23                 2040        INX   H             ;BUMP ERROR COUNTER
29D7 E3                 2050        XTHL
29D8 C3 C7 29           2060        JMP   MLOOP
29DB E1                 2070 MDONE  POP   H
29DC C9                 2080        RET
29DD                    2090 *
29DD                    2100 * BFILL FILLS THE UTILITY BUFFER=BFR WITH
29DD                    2110 * THE CHARACTER IN THE CALL ARGUMENT:
29DD                    2120 * ON EXIT, HL CONTAINS BFR'S ADDRESS.
29DD 21 AA 29           2130 BFILL  LXI   H,BFR
29E0 3E 10              2140        MVI   A,BFLEN
29E2 73                 2150 FILL   MOV   M,E
29E3 2C                 2160        INR   L
29E4 3D                 2170        DCR   A
29E5 C2 E2 29           2180        JNZ   FILL
29E8 21 AA 29           2190        LXI   H,BFR
29EB C9                 2200        RET
29EC                    2210 *
29EC                    2220 * BLOOK IS A BASIC CALLABLE FILE LOOK-UP ROUTINE.
29EC                    2230 * CALL WITH ARG=DISK DRIVE ## AFTER FILLING
29EC                    2240 * BUFFER WITH ASCII FILE NAME. ON RETURN
29EC                    2250 * HL IS A POINTER TO THE 8TH WORD OF
29EC                    2260 * THE DIRECTORY ENTRY AS PER N* DOS
29EC                    2270 * DOCUMENTATION.
29EC 7B                 2280 BLOOK  MOV   A,E
29ED 21 AA 29           2290        LXI   H,BFR
29F0 CD 1C 20           2300        CALL  DLOOK         ;RETURN POINTER
29F3 D0                 2310        RNC                 ;IF FOUND
29F4 21 00 00           2320        LXI   H,0           ;ZERO IF NOT
```

N*4

SEE PAGE H-10 FOR A PROGRAM THAT ZAPS ROMS USING THIS FEATURE.

SET THIS BYTE TO ØØ (NOP) IF YOU ONLY WANT TO VERIFY THAT THE TWO BLOCKS ARE THE SAME.

N*5

```
29F7 C9                 2330        RET
29F8                    2340 *
29F8                    2350 * FINALLY, THE FOLLOWING MAY BE USED AS AN ENTRY POINT
29F8                    2360 * TO DISABLE I/O DEVICE SELECTION FOR SOFTWARE THAT
29F8                    2370 * DOESN'T SUPPORT N*'S A-REG=DEVICE CONVENTION:
29F8 3E 00              2380 DEV0   MVI   A,0
29FA 32 5D 29           2390        STA   XIN+1
29FD 32 3D 29           2400        STA   XOUT+1
2A00                    2410        END

SYMBOL TABLE

ABC   29AA   AMS80 F000   BFILL 29DD   BFLEN 0010   BFR   29AA
BLOOK 29EC   CIN   2010   CONSB F042   CONSI F003   CONSO F009
CONTC 2016   COUT  200D   CSTS  F012   DADDR 29AD   DCOM  2455
DEV0  29F8   DLOOK 201C   DSKAD 2997   DSKRW 299C   DWRIT 201F
DXCOM 2022   ENTRY 2028   FILL  29E2   HDERR 2019   IN04  2966
IN5   2960   IN67  2963   IOASN 2978   IOCHK F015   IOCW  0003
IOSET F018   IOTBL 2919   LINES 28F8   LIST  2025   LISTB F05D
MDONE 29DB   MLOOP 29C7   MOVE  29BA   MSG   F03C   MSGXX 291B
OUT04 2955   OUT5  2946   OUT6  2949   OUT7  294C   PNCHB F051
RDRIN F006   RETRN 2995   RWCHK 202B   START 2900   TINIT 2013
TRKID 2000   VDM   FD00   VDMCW 000B   XCNTC 296C   XCOM  2989
XIN   295C   XOUT  293C
```

NOTE: IN NORTH-STAR SINGLE DENSITY RELEASE 4 DOS, THE FOLLOWING CONTROL CHARACTER PARSING ADDRESSES ARE VALID:

CNTL-C = 262DH

CANCEL = 2636H

BACK-SPACE = 263BH.

CANCEL IS NORMALLY "@", BUT YOU MAY FIND IT MORE CONVENIENT TO USE "RUBOUT" = 7FH. BACK-SPACE IS NORMALLY SHIFT-O (← = 5FH) BUT YOU MAY FIND THE ASCII. BACK-SPACE (CNTL-H = 08H) MORE CONVENIENT. TO CHANGE THESE:

```
* LF DOS 1000
* JP 0
: S 1637 40-7F (CR)
   AND/OR
: S 163C 5F-08 (CR)
 THEN
 : Z  or  G 2900
*SF DOS 1000
 * JP E800
```

DO NOT PUT THE PATCH IN THE 26xx LOCATION. DOS CONTAINS SOME SELF-MODIFYING CODE!

A 2900 1800

N*6

DOUBLE DENSITY

```
2900                    0010 * NORTH-STAR RELEASE 5 I/O DRIVERS WITH COMPATIBILITY
2900                    0020 * WITH AMS80 REV 5.8.0 AND PREVIOUS N* RELEASE 4 TO
2900                    0030 * AMS80 REV 5.7 AND 5.8 DRIVERS.    TAC  29 MAR 79
2900                    0040 *
2900                    0050 * DEVELOPED AFTER FINDING OUT WHAT NORTH-STAR REALLY
2900                    0060 * DOES IN THEIR RELEASE 5 BOOTSTRAPPING OPERATION.
2900                    0070 *
2900                    0080 * ON PAGE F-1,"PERSONALIZING . . .", NOTE THE MYSTERIOUS
2900                    0090 * MAGIC INVOLVING ADDRESSES X800 AND X900. IF YOU WERE TO
2900                    0100 * GET A MEMORY IMAGE OF DOS IN RAM WITH +LF DOS 1000 AND
2900                    0110 * LOOKED AT THE CONTENTS OF MEMORY, YOU WOULD FIND
2900                    0120 *   1000-17FF = DOS CODE AND BUFFERS
2900                    0130 *   1800-18FF = THESE I/O DRIVERS AND UTILITIES
2900                    0140 *   1900- UP  = COMMAND PARSING ROUTINES
2900                    0150 *
2900                    0160 * BUT AFTER BOOTING, YOU WILL FIND THE FOLLOWING:
2900                    0170 *   2000-28FF = DOS CODE AND BUFFERS
2900                    0180 *   2900-29FF = THESE I/O DRIVERS
2900                    0190 *   2A00-2CFF = COMMAND PARSING ROUTINES
2900                    0200 *
2900                    0210 * OR IN OTHER WORDS, THE I/O DRIVERS ARE MAGICALLY MOVED
2900                    0220 * BY 256 BYTES DURING THE BOOT-STRAPPING ROUTINE! THIS
2900                    0230 * MAKE FOR A BITCHY SITUATION IN TRYING TO ASSEMBLE
2900                    0240 * A NEW DRIVER PACKAGE AND DROVE ME UP THE WALL FOR
2900                    0250 * A WHILE! ALSO UNCOVERED ONE CODING ERROR ON N*'S
2900                    0260 * PART WHICH IS CORRECTED BELOW. TO ASSEMBLE THIS
2900                    0270 * STUFF:     LF DOS 1000   (CODE REALLY AT 2000)
2900                    0280 *       THEN IN ASMB, A 2900 1800
2900                    0290 *
2900                    0300 DOS     EQU   2000H          ;THE REAL ADDRESS
2900                    0310 GOOF    EQU   100H           ;N*'S DIDDLE OFFSET
2900                    0320         ORG   DOS+GOOF
2100                    0330 *
2100                    0340 * DOS ENTRY POINTS:
2100                    0350 *
2100                    0360 TRKID   EQU   $-GOOF
2100                    0370         DS    4              ;CURRENT TRK, EACH DRIVE
2104                    0380 *
2104                    0390         DS    3              ;=???
2107                    0400 *
2107                    0410 OFTEN   EQU   $-GOOF
2107                    0420         DS    3
210A                    0430 *
210A                    0440 START   EQU   $-GOOF
210A                    0450         DS    3
210D                    0460 *
210D                    0470 COUT    EQU   $-GOOF
210D C3 4( 29           0480         JMP   XOUT
2110                    0490 *
2110                    0500 CIN     EQU   $-GOOF
2110 C3 6( 29           0510         JMP   XIN
2113                    0520 *
2113                    0530 TINIT   EQU   $-GOOF
2113 C3 (9 29           0540         JMP   XINIT
2116                    0550 *
2116                    0560 CONTC   EQU   $-GOOF
2116 C3 7( 29           0570         JMP   XCNTC
```

(N*7)

```
2119                    0580 *
2119                    0590 HDERR  EQU   $-GOOF
2119                    0600        DS    3
211C                    0610 *
211C                    0620 DLOOK  EQU   $-GOOF
211C                    0630        DS    3
211F                    0640 *
211F                    0650 DWRIT  EQU   $-GOOF
211F                    0660        DS    3
2122                    0670 *
2122                    0680 DCOM   EQU   $-GOOF
2122                    0690        DS    3
2125                    0700 *
2125                    0710 LIST   EQU   $-GOOF
2125                    0720        DS    3
2128                    0730 *
2128                    0740 ENTRY  EQU   $-GOOF
2128                    0750        DS    3
212B                    0760 *
212B                    0770 RWCHK  EQU   $-GOOF
212B 01                 0780        DB    1             ;READ AFTER WRITE VERIFY ON
212C                    0790 *
212C                    0800 DOSER  EQU   $-GOOF
212C                    0810        DS    3
212F                    0820 *
212F                    0830 DENSE  EQU   $-GOOF
212F                    0840        DS    1
2130                    0850 *
2130                    0860 AUTO   EQU   $-GOOF
2130 01                 0870        DB    1             ;AUTO-START  OFF
2131                    0880 *
2131                    0890 INBFR  EQU   $-GOOF
2131                    0900        DS    2
2133                    0910 *
2133                    0920 PAGES  EQU   $-GOOF
2133 10                 0930        DB    16            ;SUITABLE FOR VDB4 =16 LINES/SCREEN
2134                    0940 *
2134                    0950 * CORRECTING AN ERROR I FOUND
2134                    0960 OSHIT  EQU   2291H
2134                    0970        ORG   OSHIT+GOOF
2391 CD 10 20           0980        CALL  CIN
2394                    0990 *
2394                    1000 * AMS80 ENTRY POINTS:
2394                    1010 AMS80  EQU   0F000H
2394                    1020 CONSI  EQU   AMS80+3
2394                    1030 RDRIN  EQU   AMS80+6
2394                    1040 CONSO  EQU   AMS80+9
2394                    1050 CSTS   EQU   AMS80+12H
2394                    1060 IOCHK  EQU   AMS80+15H
2394                    1070 IOSET  EQU   AMS80+18H
2394                    1080 MSG    EQU   AMS80+3CH
2394                    1090 CONSB  EQU   AMS80+42H
2394                    1100 PNCHB  EQU   AMS80+51H
2394                    1110 LISTB  EQU   AMS80+5DH
2394                    1120 IOCW   EQU   3
2394                    1130 VDMCW  EQU   0BH
2394                    1140 VDM    EQU   AMS80+0D00H
2394                    1150 *
2394                    1160 * INITIALIZATION ROUTINE = 10496D
2394                    1170 * TERMINAL INITIALIZATION ROUTINE
```

N*8

```
2394                            1180        ORG   2900H
2900                            1190 * UINIT = USER INITIALIZATION POINT -- TELLS DOS TO
2900                            1200 *  ASSUME THAT ALL FOUR DRIVES ARE ALL WHACKY:
2900 21 59 59                   1210 UINIT  LXI   H,5959H       ;59H MEANS UNDEFINED
2903 22 00 20                   1220        SHLD  TRKID
2906 22 02 20                   1230        SHLD  TRKID+2
2909 21 1C 29                   1240 XINIT  LXI   H,IOTBL       ;I/O SETUP TABLE
290C 7E                         1250        MOV   A,M
290D 32 03 00                   1260        STA   IOCW
2910 23                         1270        INX   H
2911 7E                         1280        MOV   A,M
2912 32 0B 00                   1290        STA   VDMCW
2915 23                         1300        INX   H
2916 CD 3C F0                   1310        CALL  MSG
2919 C3 28 20                   1320        JMP   ENTRY
291C                            1330 *
291C 00                         1340 IOTBL  DB    00H           ;DEFAULT IOCW ] SET THESE TO
291D 80                         1350        DB    80H           ;DEFAULT VIDCW ] MEET YOUR NEEDS
291E 0C                         1360 MSGXX  DB    0CH           ;CLEAR SCREEN
291F 4E 2A 20 44 4F             1370        DB    'N* DOS REV 5D / AMS80 REV 5.8'
     53 20 52 45 56
     20 35 44 20 2F
     20 41 4D 53 38
     30 20 52 45 56
     20 35 2E 38
293C 0A                         1380        DB    0AH           ;LINE FEED
293D 0D                         1390        DB    0DH           ;CAR RET
293E 07                         1400        DB    7             ;RING BELL (REPLACE WITH ØFFH
293F FF                         1410        DB    0FFH          ;END OF MSG  IF YOU WANT TO
                                                                  SUPRESS THE BELL)
2940                            1420 * I/O DRIVERS
2940 E6 07                      1430 XOUT   ANI   7
2942 FE 05                      1440        CPI   5
2944 FA 59 29                   1450        JM    OUT04
2947 FE 06                      1460        CPI   6
2949 78                         1470        MOV   A,B
294A FA 51 F0                   1480 OUT5   JM    PNCHB
294D CA 5D F0                   1490 OUT6   JZ    LISTB
2950 C5                         1500 OUT7   PUSH  B
2951 48                         1510        MOV   C,B
2952 CD 00 FD                   1520        CALL  VDM
2955 C1                         1530        POP   B
2956 C3 5D F0                   1540        JMP   LISTB
2959 CD 7C 29                   1550 OUT04  CALL  IOASN         ;DEVICES 0-4
295C 78                         1560        MOV   A,B
295D C3 42 F0                   1570        JMP   CONSB
2960 E6 07                      1580 XIN    ANI   7
2962 FE 05                      1590        CPI   5
2964 CA 06 F0                   1600 IN5    JZ    RDRIN
2967 F2 03 F0                   1610 IN67   JP    CONSI
296A CD 7C 29                   1620 IN04   CALL  IOASN         ;DEVICES 0-4
296D C3 03 F0                   1630        JMP   CONSI
2970                            1640 *
2970                            1650 * CNTL-C DETECTOR ALWAYS TESTS CURRENT CONSOLE!
2970 CD 12 F0                   1660 XCNTC  CALL  CSTS
2973 FE FF                      1670        CPI   0FFH          ;WAS A KEY STRUCK?
2975 C0                         1680        RNZ                 ;NOPE
2976 CD 03 F0                   1690        CALL  CONSI         ;GET THE CHARACTER
2979 FE 03                      1700        CPI   3             ;=CNTL-C?
297B C9                         1710        RET
297C                            1720 *
```

N*9

```
297C                    1730 * IOASN HANDLES I/O DEVICES (0-4) WITH:
297C                    1740 *  DEVICE 0 = CURRENT CONSOLE
297C                    1750 *  DEVICES (1-4) = AMS80 CONSOLE DEVICES (0-3)
297C                    1760 *   WHERE AFTER A CALL TO DEVICE (1-4), THE
297C                    1770 *   CURRENT CONSOLE SWITCH IN IOCW IS MODIFIED
297C                    1780 *   UNTIL A SUBSEQUENT CALL WITH A DIFFERENT
297C                    1790 *   DEVICE ID.
297C D6 01              1800 IOASN  SUI   1             ;SEPARATE 0 FROM (1-4)
297E F8                 1810        RM                  ;IF DEVICE=0
297F C5                 1820        PUSH  B
2980 4F                 1830        MOV   C,A
2981 CD 15 F0           1840        CALL  IOCHK
2984 E6 FC              1850        ANI   0FCH
2986 B1                 1860        ORA   C
2987 4F                 1870        MOV   C,A
2988 CD 18 F0           1880        CALL  IOSET
298B C1                 1890        POP   B
298C C9                 1900        RET
298D                    1910 *
298D                    1920        ORG   2997H         ;FOR COMPATIBILITY WITH
2997                    1930 *      REV 4 I/O DRIVERS DISTRIBUTED BY W3IWI
2997                    1940 *
2997                    1950 * BASIC CALLABLE DIRECT DISK READ/WRITE UTILITY
2997                    1960 *
2997                    1970 * FROM BASIC, DO 3 FILLS:
2997                    1980 *  FILL ABC  ,## 256 BYTE BLOCKS
2997                    1990 *  FILL ABC+1, 1=READ OR 0=WRITE
2997                    2000 *  FILL ABC+2, DRIVE ## (1,2 OR 3)
2997                    2010 * THEN PASS THE DISK ADDRESS WITH A CALL
2997                    2020 *  CALL DSKAD,DISK TRK/SECTOR
2997                    2030 * FINALLY DO THE DIRTY DEED WITH
2997                    2040 *  CALL DSKRW,STARTING RAM ADDRESS
2997                    2050 *
2997                    2060 * WHERE THE DECIMAL EQUIVALENTS FOR ALL THE
2997                    2070 * BASIC ENTRY POINTS AND TABLES ARE:
2997                    2080 *   DSKAD=10647 FOR SAVING DISK/DEST ADDR
2997                    2090 *   DSKRW=10652 FOR DISK/MEMORY MOVES
2997                    2100 *   ABC  =10666,7,8 FOR 3 CONTROL BYTES
2997                    2110 *   MOVE =10682 FOR MEMORY-TO-MEMORY MOVE
2997                    2120 *   BFILL=10717 TO FILL THE BFR
2997                    2130 *   BLOOK=10732 TO GET TRK/SECTOR FOR FILE
2997                    2140 *
2997                    2150 *
2997 EB                 2160 DSKAD  XCHG                ;ADR TO HL
2998 22 AD 29           2170        SHLD  DADDR
299B C9                 2180        RET
299C 21 AA 29           2190 DSKRW  LXI   H,ABC
299F 7E                 2200        MOV   A,M           ;# 256 BYTE BLOCKS
29A0 2C                 2210        INR   L
29A1 46                 2220        MOV   B,M           ;1/0 = READ/WRITE
29A2 2C                 2230        INR   L
29A3 4E                 2240        MOV   C,M           ;DRIVE ##
29A4 2A AD 29           2250        LHLD  DADDR         ;DISK TRK/SECTOR ADDR
29A7 C3 22 20           2260        JMP   DCOM
29AA                    2270 *
29AA                    2280 * UTILITY LINKAGE BUFFER AREA:
29AA                    2290 BFLEN  EQU   16
29AA                    2300 BFR    DS    BFLEN
29BA                    2310 ABC    EQU   BFR
29BA                    2320 DADDR  EQU   BFR+3
```

N*10

```
                        2330 *
                        2340 * THE FOLLOWING UTILITY IS QUITE
                        2350 * SIMILAR TO THE DISK/MEMORY UTILITY,
                        2360 * BUT HANDLES MEMORY/MEMORY TRANSFERS:
                        2370 * FIRST SET THE LENGTH WITH
                        2380 *
                        2390 *   FILL ABC,## 256 BYTE BLOCKS
                        2400 * THEN SET THE DESTINATION ADDRESS WITH:
                        2410 *   CALL DSKAD,DESTINATION STARTING ADDR
                        2420 * FINALLY, DO THE MOVE WITH
                        2430 *   CALL MOVE,SOURCE STARTING ADDR
                        2440 *
                        2450 * ON RETURN VIA HL YOU WILL GET THE
                        2460 * THE NUMBER OF ERRORS IN THE MOVE
                        2470 * THIS IS USEFUL FOR PROM ZAPPING USE.
                        2480 *
29BA 3A AA 29           2490 MOVE   LDA   ABC
29BD 47                 2500        MOV   B,A
29BE 0E 00              2510        MVI   C,0           ;BC = ## BYTES
29C0 21 00 00           2520        LXI   H,0           ;FOR
29C3 E5                 2530        PUSH  H             ;ERROR COUNTER
29C4 2A AD 29           2540        LHLD  DADDR
29C7 78                 2550 MLOOP  MOV   A,B
29C8 B1                 2560        ORA   C
29C9 CA DB 29           2570        JZ    MDONE
29CC 1A                 2580        LDAX  D
29CD 77  OR D0          2590        MOV   M,A
29CE BE                 2600        CMP   M
29CF 13                 2610        INX   D
29D0 23                 2620        INX   H
29D1 0B                 2630        DCX   B
29D2 CA C7 29           2640        JZ    MLOOP
29D5 E3                 2650        XTHL
29D6 23                 2660        INX   H             ;BUMP ERROR COUNTER
29D7 E3                 2670        XTHL
29D8 C3 C7 29           2680        JMP   MLOOP
29DB E1                 2690 MDONE  POP   H
29DC C9                 2700        RET
29DD                    2710 *
29DD                    2720 * BFILL FILLS THE UTILITY BUFFER=BFR WITH
29DD                    2730 * THE CHARACTER IN THE CALL ARGUMENT:
29DD                    2740 * ON EXIT, HL CONTAINS BFR'S ADDRESS.
29DD 21 AA 29           2750 BFILL  LXI   H,BFR
29E0 3E 10              2760        MVI   A,BFLEN
29E2 73                 2770 FILL   MOV   M,E
29E3 2C                 2780        INR   L
29E4 3D                 2790        DCR   A
29E5 C2 E2 29           2800        JNZ   FILL
29E8 21 AA 29           2810        LXI   H,BFR
29EB C9                 2820        RET
29EC                    2830 *
29EC                    2840 *
29EC                    2850 * BLOOK IS A BASIC CALLABLE FILE LOOK-UP ROUTINE.
29EC                    2860 * CALL WITH ARG=DISK DRIVE ## AFTER FILLING
29EC                    2870 * BUFFER WITH ASCII FILE NAME. ON RETURN
29EC                    2880 * HL IS A POINTER TO THE 8TH WORD OF
29EC                    2890 * THE DIRECTORY ENTRY AS PER N* DOS
29EC                    2900 * DOCUMENTATION.
29EC 7B                 2910 BLOOK  MOV   A,E
29ED 21 AA 29           2920        LXI   H,BFR
```

SEE PAGE H-10 FOR A ZAPPER PROGRAM THAT USES THIS FEATURE

(NOP) SET THIS BYTE TO Ø IF YOU ONLY WANT A VERIFY THAT THE 2 BLOCKS ARE THE SAME.

N*11

```
29F0 CD 1C 20           2930        CALL  BLOOK         ;RETURN POINTER
29F3 D0                 2940        RNC                 ;IF FOUND
29F4 21 00 00           2950        LXI   H,0           ;ZERO IF NOT
29F7 C9                 2960        RET
29F8                    2970        END
```

SYMBOL TABLE

ABC	29AA	AMS80	F000	AUTO	2030	BFILL	29DD	BFLEN	0010
BFR	29AA	BLOOK	29EC	CIN	2010	CONSB	F042	CONSI	F003
CONSO	F009	CONTC	2016	COUT	200D	CSTS	F012	DADDR	29AD
DCOM	2022	DENSE	202F	DLOOK	201C	DOS	2000	DOSER	202C
DSKAD	2997	DSKRW	299C	DWRIT	201F	ENTRY	2028	FILL	29E2
GOOF	0100	HDERR	2019	IN04	296A	IN5	2964	IN67	2967
INBFR	2031	IOASN	297C	IOCHK	F015	IOCW	0003	IOSET	F018
IOTBL	291C	LIST	2025	LISTB	F05D	MDONE	29DB	MLOOP	29C7
MOVE	29BA	MSG	F03C	MSGXX	291E	OFTEN	2007	OSHIT	2291
OUT04	2959	OUT5	294A	OUT6	294D	OUT7	2950	PAGES	2033
PNCHB	F051	RDRIN	F006	RWCHK	202B	START	200A	TINIT	2013
TRKIC	2000	UINIT	2900	VDM	FD00	VDMCW	000B	XCNTC	2970
XIN	2960	XINIT	2909	XOUT	2940				

VDB-4

A>ASM VDB4.BZX
CP/M ASSEMBLER - VER 1.4

V-1

```
        TITLE  'VDB4'
        ; VIDEO DISPLAY DRIVER FOR SSM VB1-B BOARD
;
;    ************ UPDATED FOR AMS80 REV 5.8 ************
;-------------------------------------------------------------;
;                                                             ;
;   THIS VIDEO DRIVER CRAMS INTO 512 BYTES OF RAM ALL THE     ;
;   NIFTY FEATURES THAT WE COULD THINK OF. THE DRIVER USES    ;
;   ONE BYTE OF RAM IN THE 'BASE-PAGE' OF 8080 MEMORY:        ;
;                                                             ;
;          VIDEO CONTROL BYTE AT 0008H                        ;
; +------+------+------+------+------+------+------+------+  ;
; ! 80H  ! 40H  ! 20H  ! 10H  ! 08H  ! 04H  ! 02H  ! 01H  !  ;
; !      !      !      !      !      !      !      !      !  ;
; ! <---MODE--> ! HOLD ! NOT  ! CLOCK! HNSTY!<-PROTECTED->!!;
; !   CONTROL   ! BIT  ! NOW  !ON/OFF! FLAG ! LINE COUNT  !  ;
; !     BITS    !      ! USED ! BIT  ! BIT  !    BIITS    !  ;
; !             !      !      !      !      !             !  ;
; +------+------+------+------+------+------+------+------+  ;
;                                                             ;
;  BITS 80H+40H CONTROL VIDEO MODE AS FOLLOWS:                ;
;      MODE-0 = 00H = PAGE MODE WITH LINE OVERFLOW            ;
;      MODE-1 = 40H = PAGE MODE WITH LINE TRUNCATIONN         ;
;      MODE-2 = 80H = SCROLL MODE WITH LINE OVERFLOW          ;
;      MODE-3 = C0H = SCROLL MODE WITH LINE TRUNCATION        ;
;  WHERE OVERFLOW CONTINUES WITH CHARACTERS PAST #64          ;
;     WRAPPING AROUND AND APPEARING ON NEXT LINE              ;
;  AND WHERE TRUNCATION MAKES ALL CHARACTERS PAST #64         ;
;             WRITE OVER #64 .                                ;
;  THESE TWO BITS CAN BE SET BY THE 'CV' COMMAND IN AMS80.    ;
;                                                             ;
;  IN EITHER PAGE MODE, OUTPUT WILL PAUSE AT THE BOTTOM OF    ;
;     THE SCREEN WAITING FOR OPERATOR INTERVENTION VIA THE    ;
;      CONSOLE. IF THE [SPACE-BAR] IS DEPRESSED, THE SCREEN   ;
;     WILL BE ERASED AND THE OUTPUT WILL RESUME AT THE TOP.   ;
;     IF THE SLASH [/] BAR IS PRESSED, THE OUTPUT WILL        ;
;     SCROLL UP ONE LINE AND AGAIN WAIT FOR INPUT.            ;
;     IF THE MINUS-SIGN [-] IS PRESSED, THE MODE CONTROL      ;
;     WILL REVERT TO THE SCROLL MODE (E.G.  80H BIT ET).      ;
;     IF ANY OTHER KEY IS PRESSED, THE SCREEN WILL CLEAR      ;
;     THE SAME AS WITH THE [SPACE-BAR], BUT THE  'HLD BIT'    ;
;     = 20H WILL BE SET SO THAT THE CONSOLE I/O ROUTINE       ;
;     WILL COME BACK AND RE-READ THE HARDWARE CHARACTER        ;
;     BUFFER THE NEXT TIME THRU. ERGO, THIS BECOMES THE       ;
;     NEXT INPUT CARACTER.                                    ;
;                                                             ;
;  TO USE THE DRIVER, PUT THE ASCII CHARACTER TO BE WRITTEN   ;
;  IN THE <C> REGISTER. ALL REGISTERS ARE PRESERVED. STACK    ;
;  REQUIREMENTS VARY, BUT YOU BETTER HAVE AROUND 20 BYTES     ;
;  FOR SAFETY.                                                ;
;                                                             ;
;  THE 'HNSTY' (HONESTY) BIT (04H):                           ;
;     IF THIS BIT = 0 (SUGGESTED DEFAULT), THEN THESE         ;
;     SPECIAL CONTROL FUNCTIONS ARE ENABLED:                  ;
;         CNTL-U = 15H = MOVE CURSOR UP                       ;
;         CNTL-D = 04H = MOVE CURSOR DOWN                     ;
;         CNTL-F = 06H = MOVE CURSOR RIGHT                    ;
```

```
;        CNTL-H = 08H = MOVE CURSOR LEFT                        ;
;       SHIFT-O = 5FH = MOVE CURSOR LEFT                        ;
;        CNTL-G = 07H = RING THE SYSTEM BELL                    ;
;        CNTL-A = 01H = HOME (UPPER-LEFT CORNER) CURSOR         ;
;        CNTL-I = 09H = TAB TO NEXT EIGHT'S COLUMN              ;
;                                                               ;
;    REGARDLESS OF THE HONESTY BIT, THE FOLLOWING CONTROL       ;
;    FUNCTIONS ARE ALWAYS ENABLED:                              ;
;        CNTL-K = 0BH = VERT. TAB. CLEARS SCREEN TO THE         ;
;                       TOP OF THE PROTECTED MEMORY AREA.       ;
;        CNTL-J = 0AH = LINE FEED. CLEAR THE REST OF THE        ;
;                       CURRENT LINE AND ADVANCE TO THE         ;
;                       START OF THE NEXT LINE.                 ;
;        CNTL-M = 0DH = CAR.RET. MOVE CURSOR TO LEFT            ;
;                           MARGIN OF CURRENT LINE.             ;
;        CNTL-L = 0CH = FORM FEED = CLEAR ENTIRE SCREEN,        ;
;                       AND SET CURSOR TO TOP-LEFT CORNER       ;
;                       OF SCREEN, REGARDLESS OF ANY            ;
;                       PROTECTED LINES.                        ;
;                                                               ;
;  BITS 01H+02H ARE THE COUNT USED TO INDICATE THE NUMBER       ;
;     OF PROTECTED LINES AT THE TOP OF THE SCREEN, UP TO A      ;
;     MAXIMUM OF 3. PROTECTED LINES ARE SAVED UNDER PAGE        ;
;     CLEAR (E.G. CNTL-K) OR SCROLLING OPERATIONS, ALTHOUGH     ;
;     THEY MAY BE ERASED BY THE VERT TAB [CNTL-L] COMMAND.      ;
;     AFTER BEING CLEARED, THE CURSOR IS IN THE PROTECTED       ;
;     AREA (E.G. AT THE TOP-LEFT CORNER OF THE SCREEN), SO      ;
;     YOU CAN THEN WRITE TO THEM IN THE NORMAL MANNER THRU      ;
;     CALLS TO THE VDB-4 DRIVER. THESE PROTECTED LINES ARE      ;
;     INTENDED FOR PAGE HEADING INFO OR FOR 'PERMANENT'         ;
;     DISPLAYS LIKE A TIME-OF-DAY CLOCK. NOTE THAT THE          ;
;     GRAPHICS PORTION OF VDB4 ALSO RESPECTS THESE PROT-        ;
;     ECTED LINES AND  ILL ONLY PLOT IN UN-PROTECTED AREAS.     ;
;                                                               ;
;  AS PER AMS80 ASSIGNMENT STANDARDS, THE 1K-BYTE OF            ;
;  VIDEO RAM IS ASSUMED TO LIE BETWEEN CC00-CFFF. THIS          ;
;  ASSIGNMENT IS ALSO COMPATIBLE WITH PROCESSOR TECHNOLOGY      ;
;  SOL ASSIGNMENTS.                                             ;
;                                                               ;
;---------------------------------------------------------------;
;                                                               ;
;                                                               ;
;   THE CLOCK ON / OFF BIT (08H) IS RESERVED FOR USE BY         ;
;   A REAL-TIME TIME-OF-DAY CLOCK TO INDICATE WHETHER IT        ;
;   IS TO BE DISPLAYED IN THE UPPER-RIGHT HAND CORNER OF        ;
;   THE VIDEO DISPLAY ( 0=ON, 1=OFF ).                          ;
;                                                               ;
;                                                               ;
;   NOTES ON INTERNAL CUSTOMIZATION.                            ;
;   YOUR SYSTEM BELL:                                           ;
;                          THE DEFAULT SOFTWARE IGNORES IT!     ;
;   YOU SHOULD MODIFY THE CALL ADDRESS AT FD83H TO AGREE        ;
;   WITH YOUR SYSTEM-DEPENDENT BELL DRIVER.                     ;
;                                                               ;
;   IF YOU SHOULD DESIRE TO DELETE OR CHANGE A CONTROL          ;
;   CHARACTER IN THE CODE BETWEEN FD37H AND FD80H, THE MOST     ;
;   DIRECT WAY IS TO CHANGE THE STATEMENT FROM:                 ;
;               CPI <OLD CONTROL CHARACTER>                     ;
;   TO EITHER:  CPI <NEW CONTROL CHARACTER>  [TO CHANGE IT]     ;
;   OR TO    :  CPI <LINE FEED = 0AH>        [TO DELETE IT]     ;
```

```
;   THIS WORKS TO DELETE A CHARACTER BECAUSE THE <LINE FEED>;
;   HAS ALREADY BEEN HANDLED AT FD24H AND CANNOT BE DETECTED;
;   AGAIN. THE USE OF TWO DIFFERENT BACK-SPACE CHARACTERS   ;
;   <CNTL-H = 08H> AND <SHIFT-O/UNDERSCORE/BACK-ARROW = 5FH>;
;   WAS INCLUDED BECAUSE BOTH HAVE FOUND THEIR WAY INTO     ;
;   COMMON USAGE. FOR EXAMPLE, ALL THE NORTH-STAR SOFTWARE  ;
;   AND MUCH OF THE SOFTWARE THAT WORKS WITHIN A NORTH-STAR ;
;   ENVIRONMENT, USES THE <SHIFT-O> CONVENTION, WHILE ASCII ;
;   TERMINALS TEND TO USE THE <CNTL-H> CONVENTION. IT IS    ;
;   SUGGESTED THAT YOU LEAVE BOTH DEFINITIONS ENABLED UNLESS;
;   YOU RUN INTO A VERY SERIOUS CONFLICT PROBLEM.           ;
;                                                           ;
;-----------------------------------------------------------;
;                                                           ;
;  INFO ON GRAPHICS PACKAGE:                                ;
;  THE GRAPHICS DRIVER SOFTWARE WORKS IN A POINT-PLOTTING   ;
;  MODE. YOU GIVE THE SOFTWARE THE X (HORIZONTAL) AND Y     ;
;  (VERTICAL) COORDINATES OF A DOT, AND DEPENDING ON THE    ;
;  ROUTINE CALLED, A WHITE OR BLACK DOT IS PLOTTED OR THE   ;
;  STATUS OF THE DOT IS RETURNED. THE DOT CELLS OCCUPY THE  ;
;  SAME PHYSICAL AREA AS A CHARACTER, WITH 3 VERT x 2 HORIZ ;
;  DOTS PER CHARACTER CELL. THEREFORE, GIVEN A 16 VERT LINE ;
;  x 64 HORIZ CHARACTER SCREEN AREA, THERE ARE 48 x 128     ;
;  GRAPHICS CELLS ON THE SCREEN. WHEN CALLED WITH AN ADDRESS;
;  X=00, Y=00 YOU ADDRESS THE LOWER LEFT-HAND CORNER OF THE ;
;  SCREEN. X=127, Y=00 IS THE LOWER RIGHT-HAND CORNER, ETC. ;
;  IF YOU ASK FOR AN X ADDRESS >=128, THE REQUESTED ADDRESS ;
;  IS TAKEN MODULO 128. IN THE Y (VERTICAL) DIRECTION, THE  ;
;  MAXIMUM NUMBER OF CELLS =47 IF THE SCREEN IS UN-PROTECTED;
;  AND IS 47-3*N, WHERE N IS THE NUMBER OF PROTECTED LINES  ;
;  (0 <= N <= 3). IF YOU REQUEST A VERTICAL ADDRESS LARGER  ;
;  THAN 47-3*N, THE VERTICAL ADDRESS IS TAKEN MODULO 48-3*N.;
;  THESE MODULO OPERATIONS FORCE GRAPHICS TO "WRAP AROUND"  ;
;  SO THAT IF, FOR EXAMPLE, YOU TRY TO PLOT ABOVE THE TOP   ;
;  BORDER, THE DATA WILL NOT BE LOST, BUT WILL APPEAR AT THE;
;  BOTTOM OF THE SCREEN.                                    ;
;                                                           ;
;  THE NORMAL ASSEMBLY LANGUAGE ENTRIES TO THE GRAPHICS     ;
;  PACKAGE ARE:                                             ;
; WHITE: FE65H = 65125.DEC = PLOT WHITE DOT                 ;
; BLACK: FE6BH = 65131.DEC = PLOT BLACK DOT                 ;
; CHECK: FE60H = 65120.DEC = IS THE CELL WHITE OR BLACK?    ;
; CNVTX: FE70H = 65136.DEC = GET CELL ADDRESS. IF IT IS A   ;
;                            TEXT CHARACTER, REPLACE IT WITH ;
;                            A GRAPHICS BLANK CHARACTER.    ;
; CBLNK: FE7AH = 65146.DEC = BLANK THE ENTIRE 2x3 CELL      ;
;     THESE ROUTINES ARE CALLED WITH THE ADDRESS IN <HL>    ;
;        WHERE <H> = VERTICAL (Y) COORDINATE                ;
;          AND <L> = HORIZONTAL (X) COORDINATE              ;
;                                                           ;
;  IF EITHER OF THESE ADDRESSES OVERFLOW THE SCREEN AND     ;
;  HAD TO BE MOD'D WITH (128) OR (48-3*N), ON EXIT THE NEW  ;
;  VALUES ARE IN <HL>, OTHERWISE <HL> IS UNCHANGED.         ;
;                                                           ;
;     ON EXIT:  <DE> = MEMORY ADDRESS FOR THE CELLL         ;
;               <HL> = SCREEN COORDINATES.                  ;
;               <C>  = BIT MASK FOR THE CELL.               ;
;               <B>  = UNCHANGED.                           ;
;               <A>  = CONTENTS OF <DE> REGISTER PAIR FROM  ;
;                  [WHITE]\[BLACK]\[CNVTX]\[CBLNK] ROUTINES ;
```

```
;              <A>  = ZERO FROM [CHECK] IF CELL IS WHITE    ;
;              <A>  = <C>  FROM [CHECK] IF CELL IS BLACK    ;
;                                                           ;
;-------------------------- ------------------------------;
;                                                           ;
;  GRAPHICS INTERFACES FROM BASIC ARE SIMILAR TO THOSE      ;
;  DESCRIBED ABOVE. THESE INTERFACES HAVE BEEN TESTED WITH  ;
;  NORTH-STAR AND PROCESSOR TECHNOLOGY SOFTTARE WHICH USE   ;
;  THE FOLLOWING ASSEMBLY LANGUAGE CALLING CONVENTIONS:     ;
;           R = CALL ( ADDRESS , DATA )                     ;
;                     OR                                    ;
;    X = ARG ( DATA ) FOLLOWED BY  R = CALL ( ADDRESS )     ;
;                                                           ;
;     WHERE: ASSEMBLY LANGUAGE ROUTINE STARTS AT (ADDRESS)  ;
;            AND (DATA) IS PASSED IN THE <DE> REGISTER PAIR ;
;            AND <HL> REGISTER PAIR CONVEYS (R) AS THE DATA ;
;            RETURNED TO BASIC FROM THE EXTERNAL ROUTINE.   ;
;                                                           ;
;  THE (DATA) ARGUMENT IN <DE> = 256*Y + X TO PACK THE TWO  ;
;  ADDRESSES INTO ONE VARIABLE.  OUTINES AVAILABLE ARE:     ;
;   BASW:  FEEFH = 65263.DEC = PLOT WHITE DOT AT (Y,X)      ;
;   BASB:  FEF4H = 65268.DEC = PLOT BLACK DOT AT (Y,X)      ;
;                              BOTH [BASW] AND [BASB] RETURN ;
;                              (R) = 256*Y + X SCREEN COORDS ;
;                              AFTER BEING MODULO'D TO KEEP  ;
;                              THEM ON THE SCREEN.           ;
;   BASA:  FEF9H = 65273.DEC = GET SCREEN MEMORY ADDRESS     ;
;                              FOR (Y,X) AND RETURN TO USER  ;
;   BASC:  FEDDH = 65245.DEC = CHECK CELL CONTENTS. IF CELL  ;
;                              IS OCCUPIED BY AN ASCII(TEXT) ;
;                              CHARACTER, RETURN IT AS (R).  ;
;                              IF CELL IS WHITE, (R) = 255   ;
;                              IF CELL IS BLACK, (R) = ZERO  ;
;                                                           ;
;-----------------------------------------------------------;
;                                                           ;
;  THIS VDM DRIVER IS DESIGNED SPECIFICALLY FOR SSM (FORMERLY ;
;  SOLID STATE MUSIC) VB1 AND VB1B BOARDS, BUT SHOULD WORK JUST ;
;  AS WELL WITH THE POLYMORPHICS* BOARD. THIS SOFTWARE WAS   ;
;  WRITTEN AND RE-WRITTEN MANY TIMES CIRCA 1977 AND 1978,   ;
;  BUILDING ON IDEAS IN THE CYBERCOM VB-1 SOFTWARE PACKAGE  ;
;  AND ON IDEAS FROM JOE KASSER. THIS REVISION MARCH, 1979 ;
;   BY TOM CLARK, W3IWI, CORRECTS ONE ERROR & UPDATES FOR REV 5B. ;
;-----------------------------------------------------------;
;  MEMORY DEFINITION CONSTANTS
;
FD00 =          VDROM   EQU     0FD00H  ;ROM LOCATION FOR ASSEMBLY
CC00 =          VID     EQU     0CC00H   ;VIDEO RAM LOCATION
00CC =          VIDHI   EQU     0CCH     ;HIGH BYTE OF VIDEO ADDR
00D0 =          VDEND   EQU     VIDHI+4 ;HIGH BYTE OF FIRST NON-VIDEO AREA
;
;
;          SET ASIDE 4 BYTES IN BASE-PAGE FOR AMS80 WORKING STORAGE:
000E =          VDPTR    EQU    0EH     ;CURSOR POINTER MEMORY LOCATION
000B =          VDMCW    EQU    0BH     ;MODE CONTROL BYTE
;
;          OTHER CONSTANTS:
00FF =          GBLNK   EQU     0FFH    ;GRAPHICS BLANK CHARACTER
0020 =          BLANK   EQU     20H     ;ASCII SPACE (BLANK) CHARACTER
00AD =          CURSE   EQU     0ADH    ;CURSOR CHARACTER FROM GRAPHICS SET
```

FIRST 3 BYTES OF RAM ADDRESSED BY 000E-F ARE USED BY VDB-4 FOR RUNNING IT'S CURSOR.

* OR PROCESSOR TECHNOLOGY OR ITHACA AUDIO BOARDS, ALTHOUGH THESE BOARDS DO NOT SUPPORT GRAPHICS.

```
                        ;NOTE: CURSE IS ALSO A REVERSE-VIDEO MINUS SIGN IF NOT IN GRAPHICS.
                        ;
                        ;               ENTRY POINTS TO AM380:
F003 =                  CONSI   EQU     0F003H  ;INTERFACE TO   AM380
                        ;
                                ;NON-DISPLAYABLE CHARACTERS FOR CONTROL:
000C =                  FF      EQU     0CH      ;CNTL-L = FORM FEED = CLEAR ENTIRE SCREEN
                                                 ;INCLUDING THE PROTECTED AREA.
000A =                  LF      EQU     0AH      ;CNTL-J = LINE FEED = CLEAR & ADVANCE LINE
000B =                  TOP     EQU     0BH     ;CNTL-K = CLEAR TO TOP OF UN-PROTECTED
                                                ;SCREEN AREA, BUT LEAVE PROTECTED LINES.
000D =                  CR      EQU     0DH     ;CNTL-M = CAR RET = MOVE TO LEFT MARGIN
0015 =                  UP      EQU     15H     ;CNTL-U = MOVE UP
0004 =                  DOWN    EQU     04H     ;CNTL-D = MOVE DOWN
0006 =                  RIGHT   EQU     06H     ;CNTL-F = MOVE FWD (RIGHT)
0008 =                  LEFT1   EQU     08H     ;CNTL-H = MOVE BACK (LEFT)
005F =                  LEFT2   EQU     5FH     ;SHIFT-O = PACK ARROW ALSO = MOVE LEFT
0001 =                  HOME    EQU     01H     ;CNTL-A = MOVE HOME (UPPER LEFT CORNER)
                                               ;OF UNPROTECTED AREA OF SCREEN.
0009 =                  HTAB    EQU     09H      ;CNTL-I = PRESET TAB TO NEXT EIGHTH COLUMN
0007 =                  BELL    EQU     07H      ;CNTL-G = BELL CHARACTER
                        ;
                        ;-----------------------------------------------------------
                        ;
FD00                            ORG VDROM ;TO START ASSEMBLY AT  FD ØØH
                        ;
                        ;-----------------------------------------------------------
                        ; ----- BOOB-TUBE TTY-LIKE ENTRY POINT, WITH DATA IN <C> -----
FD00 E5                 VDTTY:  PUSH    H       ;SAVE ALL
FD01 D5                         PUSH    D       ;  REGISTERS TO KEEP
FD02 C5                         PUSH    B       ;    FRIENDLY NEIGHBORS!
FD03 F5                         PUSH    PSW
FD04 2A0E00                     LHLD    VDPTR   ;=CURRENT CURSOR ADDRESS
FD07 5E                         MOV     E,M
FD08 23                         INX     H       ;NOW MAKE SURE THAT CURSOR IS IN
FD09 7E                         MOV     A,M     ;THE REAL VDM AREA, NOT NO-MAN'S LAND.
FD0A E603                       ANI     3       ;     SORRY --- NO PERSON'S LAND.
FD0C C6CC                       ADI     VIDHI
FD0E 57                         MOV     D,A
FD0F 23                         INX     H
FD10 46                         MOV     B,M
FD11 EB                         XCHG
FD12 70                         MOV     M,B
                        ;IDENTIFY INPUT CHARACTER AND SEE IF IT IS A CONTROL CHARACTER
FD13 79                         MOV     A,C
FD14 FE0C               CFF:    CPI     FF      ;=CNTL-L <CLEAR TO TOP OF SCREEN>?
FD16 C21FFD                     JNZ     CCR
FD19 2100CC                     LXI     H,VID
FD1C C3C1FD                     JMP     CLRHL
FD1F FE0D               CCR:    CPI     CR      ;=CNTL-M <CAR.RET.>?
FD21 CACCFD                     JZ       VIDCR
FD24 FE0A               CLF:    CPI     LF      ;=CNTL-J <LINE FEED>?
FD26 CAD3FD                     JZ       VIDLF
FD29 FE0B               CTOP:   CPI     TOP     ;=CNTL-K <CLEAR TO TOP OF UN-PROTECTED AREA>?
FD2B CABEFD                     JZ      VDTOP
FD2E 3A0B00             HNSTY:  LDA     VDMCW   ;TEST IF "HONESTY"
FD31 E604                       ANI     04H     ;BIT IS SET--
FD33 79                         MOV     A,C     ;BUT FIRST, RESTORE DATA FOR FUTURE USE
FD34 C289FD                     JNZ     DSPLY   ;BYPASS FURTHER CHARACTER TESTING TO BE HONEST
                                                ;CHECK  FOR CURSOR CONTROL CHARACTERS:
```

```
FD37 FE15      CUP:    CPI     UP        ;=CNTL-U?
FD39 C242FD            JNZ     CDOWN
FD3C 01C0FF            LXI     B,-64
FD3F C3A9FD            JMP     CRADJ
FD42 FE04      CDOWN:  CPI     DOWN      ;=CNTL-D?
FD44 C24DFD            JNZ     CLEFT
FD47 014000            LXI     B,64
FD4A C3A9FD            JMP     CRADJ
FD4D FE08      CLEFT:  CPI     LEFT1     ;= CNTL-H?
FD4F CA57FD            JZ      XLEFT
FD52 FE5F              CPI     LEFT2     ;= LEFT-ARROW (SHIFT-O)?
FD54 C25DFD            JNZ     CRITE
FD57 01FFFF    XLEFT:  LXI     B,-1
FD5A C3A9FD            JMP     CRADJ
FD5D FE06      CRITE:  CPI     RIGHT     ;=CNTL-F?
FD5F CAA6FD            JZ      CRRT
FD62 FE01      CHOME:  CPI     HOME      ;=CNTL-A?
FD64 C26DFD            JNZ     CTAB
FD67 CD55FE            CALL    PRTOP
FD6A C3AAFD            JMP     VIDRT
FD6D FE09      CTAB:   CPI     HTAB      ;=CNTL-I?
FD6F C27EFD            JNZ     CBELL
FD72 3620      TAB:    MVI     M,BLANK
FD74 23                INX     H
FD75 7D                MOV     A,L
FD76 E607              ANI     7
FD78 C272FD            JNZ     TAB
FD7B C3AAFD            JMP     VIDRT
FD7E FE07      CBELL:  CPI     BELL
FD80 C289FD            JNZ     DSPLY

FD83 00        DING:   NOP               ;EXTERNAL BELL ENTRY POINT
                                         ;CALL YOUR ROUTINE
FD84 00                NOP               ;*****PATCH HERE ***
FD85 00                NOP
FD86 C3AAFD            JMP VIDRT
               ;----   SURVIVORS ARE DISPLAYABLE
FD89 7D        DSPLY:  MOV     A,L       ;CHECK
FD8A E63F              ANI     3FH       ; FOR
FD8C FE3F              CPI     3FH               ;  LINE
FD8E C2A5FD            JNZ     VIDB0             ;    ENDING
               ;IGNORE CHARACTER IF END OF LINE AND MODE IS ODD
FD91 3A0B00            LDA     VDMCW     ;GET MODE BYTE
FD94 E640              ANI     40H
FD96 C2AAFD            JNZ     VIDRT
FD99 D5                PUSH    D
FD9A 23                INX     H
FD9B 7C                MOV     A,H
FD9C FED0              CPI     VDEND
FD9E CADEFD            JZ      VDLF2     ;OUT OF SCREEN
FDA1 CD35FE            CALL    VDLF6     ;BLANK LINE
FDA4 D1                POP     D
FDA5 71        VIDB0:  MOV     M,C
FDA6 010100    CRRT:   LXI     B,1       ;MOVE   CURSOR RIGHT ONE SPACE
                                         ;ADJUST CURSOR POINTER
FDA9 09        CRADJ:  DAD     B
                                         ;COMMON EXIT CODE-- NORMALIZE CURSOR POINTER
FDAA 7C        VIDRT:  MOV     A,H
FDAB E603              ANI     3
FDAD C6CC              ADI     VIDHI
```

NOTE:

- IF FD83-FD85 ARE SET TO 00 00 00 THEN THE BELL DOES NOTHING.
- IF FD83-FD8F ARE JMP DSPLY (C3 89 FD) THEN A BELL CHARACTER (e.g. Θ OR ⌂ depending on your video board's character generator chip) IS SHOWN ON THE CRT.
- IF YOU PATCH IN A CALL TO A "REAL" BELL ROUTINE, AND DON'T CLOBBER THE STACK IT SHOULD DING ON COMMAND, SEE PAGE C-9, LINES 1980-2160 FOR AN EXAMPLE.

```
FDAF 67                 MOV     H,A
FDB0 7E                 MOV     A,M
FDB1 36AD               MVI     M,CURSE
FDB3 EB                 XCHG
FDB4 77                 MOV     M,A
FDB5 2B                 DCX     H
FDB6 72                 MOV     M,D
FDB7 2B                 DCX     H
FDB8 73                 MOV     M,E
                                        ;RESTORE ALL REGISTERS AND QUIT
FDB9 F1                 POP     PSW
FDBA C1                 POP     B
FDBB D1                 POP     D
FDBC E1                 POP     H
FDBD C9                 RET     ;--DONE!

                                        ;PROCESS FORM FEED,
                                        ; CLEAR SCREEN AND HOME CURSOR
FDBE CD55FE     VDTOP:  CALL    PRTOP
FDC1 E5         CLRHL:  PUSH    H
FDC2 3620       VIDFC:  MVI     M,BLANK
FDC4 23                 INX     H
FDC5 7C                 MOV     A,H
FDC6 FED0               CPI     VDEND
FDC8 FAC2FD             JM      VIDFC
FDCB E1                 POP     H
                                        ;PROCESS CARRIAGE RETURN
FDCC 7D         VIDCR:  MOV     A,L
FDCD E6C0               ANI     0C0H
FDCF 6F                 MOV     L,A
FDD0 C3AAFD             JMP     VIDRT
                                        ;LINE FEED--CLEAR CURRENT LINE, THEN ADVANCE
FDD3 D5         VIDLF:  PUSH    D
FDD4 114000             LXI     D,40H
FDD7 19                 DAD     D
FDD8 7C                 MOV     A,H
FDD9 FED0               CPI     VDEND
FDDB C245FE             JNZ     VDLF3
                                        ;TEST FOR SCROLL/FULL SCREEN MODE
FDDE 3A0B00     VDLF2:  LDA     VDMCW   ;GET CONTROL BYTE
FDE1 17                 RAL
FDE2 DA0DFE             JC      VDSCL
FDE5 21FFCF             LXI     H,VID+1023      ;POINT TO LAST CHARACTER ON SCREEN
FDE8 3680               MVI     M,80H   ;SET IT TO FULL GRAPHIC WHITE OR INVERTED GREEK ALPH
A
FDEA CDF1FD     VDLF9:  CALL    BRKIN   ;WAIT FOR KEYBOARD OK
FDED D1                 POP     D       ;SCRUFF A PUSH!
FDEE C3BEFD             JMP     VDTOP
FDF1 CD03F0     BRKIN:  CALL    CONSI   ;AND WAIT FOR A KEY TO BE PRESSED
FDF4 FE20               CPI     BLANK   ;= <SPACE>?
FDF6 C8                 RZ
FDF7 FE2F       SLASH:  CPI     '/'     ;=SLASH?
FDF9 CA09FE             JZ      SCRL1
FDFC FE2D       MINUS:  CPI     '-'     ;=MINUS?
FDFE C24CFE             JNZ     SAVER   ;TO FLAG CHARACTER-IN-HOLD
FE01 3A0B00             LDA     VDMCW   ;SET TO
FE04 F680               ORI     80H     ; SCROLL
FE06 320B00             STA     VDMCW   ;  MODE
FE09 D1         SCRL1:  POP     D
FE0A 3620               MVI     M,BLANK
```

```
FE0C 23                 INX     H
                                        ;---SCROLL MODE ENTRY
FE0D E5         VDSCL:  PUSH    H       ;SET    UP FOR SCROLL
FE0E CD55FE             CALL    PRTOP   ;ADDRESS OF TOP OF SCREEN
FE11 EB                 XCHG            ;<DE>=TOP OF UNPROTECTED MEMORY
FE12 214000             LXI     H,40H   ;ADDRESS OF NEXT LINE
FE15 19                 DAD     D       ;<HL>=SECOND LINE OF UNPROTECTED MEMORY
FE16 7E         VDS1:   MOV     A,M     ;SCROLL LOOP
FE17 12                 STAX    D
FE18 13                 INX     D
FE19 23                 INX     H
FE1A 7C                 MOV     A,H
FE1B FED0               CPI     VDEND
FE1D C216FE             JNZ     VDS1
FE20 21C0CF             LXI     H,VID+960       ;POINT TO BOTTOM LINE
FE23 CD35FE             CALL    VDLF6   ;BLANK IT
FE26 E1                 POP     H
FE27 11C0FF             LXI     D,-40H
FE2A 19                 DAD     D       ;BACK UP TO LAST  POSITION
FE2B D1                 POP     D
FE2C 79         VIDF1:  MOV     A,C     ;GET CHARACTER
FE2D FE20               CPI     BLANK   ;TEST FOR PRINTING
FE2F FAAAFD             JM      VIDRT   ;IGNORE IT
FE32 C3A5FD             JMP     VIDB0   ;PRINT CHARACTER ON NEW LINE
                VDLF6:                  ;SUBROUTINE TO FILL NEW LINE WITH BLANKS
FE35 E5                 PUSH    H
FE36 7D                 MOV     A,L
FE37 E6C0               ANI     0C0H
FE39 6F                 MOV     L,A
FE3A 3620       VDLF4:  MVI     M,BLANK
FE3C 23                 INX     H
FE3D 7D                 MOV     A,L
FE3E E63F               ANI     3FH     ;DONE WITH THE LINE?
FE40 C23AFE             JNZ     VDLF4
FE43 E1                 POP     H
FE44 C9                 RET
FE45 CD35FE     VDLF3:  CALL    VDLF6   ;FILL LINE WITH BLANKS
FE48 D1                 POP     D
FE49 C3AAFD             JMP     VIDRT

                ;ROUTINE TO INDICATE THAT THE CONSOLE INPUT
                ;PORT HAS A CHARACTER-IN-HOLD IN ITS HARDWARE INPUT
                ;REGISTER. CONSOLE INPUT DRIVERS SHOULD ALL EXAMINE
                ;THE 20H BIT IN VDMCW IN ADDITION TO TESTING THEIR
                ;STATUS PORT TO READ INPUT DATA. AN EXAMPLE OF THIS
                ;USAGE IS SHOWN IN THE EXAMPLE IN SECTION C OF THE
                ;AMS80 REV 5.8 DOCUMENTATION.
FE4C 3A0B00     SAVER:  LDA     VDMCW
FE4F F620               ORI     20H
FE51 320B00             STA     VDMCW
FE54 C9                 RET
                ;
                ;ROUTINE TO EXTRACT THE CURRENT UN-PROTECTED TOP
                ;OF MEMORY, AS INDICATED BY THE LOW 2 BITS OF VDMCW
                ;       NOTE THAT THIS ROUTINE COULD BE CALLED FROM THE OUTSIDE
                ;       WORLD (INCLUDING FROM BASIC) TO GET INFO ON PROTECT STATUS.
                ;
FE55 26CC       PRTOP:  MVI     H,VIDHI
FE57 3A0B00             LDA     VDMCW
FE5A 0F                 RRC
```

? • SEE PAGES C-4, C-11/12 INVOLVING HOLDST/MSKCL ROUTINES.

```
FE5B 0F                 RRC
FE5C E6C0               ANI 0C0H
FE5E 6F                 MOV L,A
FE5F C9                 RET
                ;-----------------------------------------------------------;
                ;       GRAPHICS POINT-PLOT DRIVER ROUTINE FOR SSM VB-1,VBI-B
                ;       OR POLYMORPHICS VIDEO BOARDS:
                ;
                ;       ON ENTRY, <HL> CONTAINS THE DESIRED COORDINATE PAIR,
                ;       WITH <H>=Y-COORD, <L>=X-COORD, WITH <0,0>=LOWER-LEFT CORNER
                ;
                ;       ON EXIT, <A> CONTAINS INFO DESCRIBED BELOW
                ;                <B> IS NOT CHANGED
                ;                <C> IS BIT MASK FOR SPECIFIED CELL
                ;                <DE> IS MEMORY ADDRESS FOR SPECIFIED CELL
                ;                <HL> IS STILL COORDINATE ADDRESS
                ;
                ;-----------------------------------------------------------;
                ;
FE60 CD70FE     CHECK:  CALL    CNVTX   ;IS CELL OCCUPIED?
FE63 A1                 ANA     C
FE64 C9                 RET
                                        ;WHITE TURNS CELL ON. <A>=MEMORY CONTENTS
FE65 CD7AFE     WHITE:  CALL    CBLNK   ;CONVERT AND BLANK
FE68 A9                 XRA     C       ;NOW MAKE IT WHITE
FE69 12                 STAX    D
FE6A C9                 RET
                                        ;BLACK TURNS CELL OFF. <A>=MEMORY CONTENTS
FE6B CD7AFE     BLACK:  CALL    CBLNK
FE6E 12                 STAX    D
FE6F C9                 RET
                                        ;CNVTX CALLS CNVRT TO GET ADDRESS.
                                        ; IF THAT MEMORY CELL IS OCCUPIED BY
                                        ; A TEXT CHARACTER, IT IS CONVERTED
                                        ;INTO A VIDEO GRAPHICS BLANK.
FE70 CD83FE     CNVTX:  CALL    CNVRT
FE73 FE80               CPI     80H     ;=TEXT?
FE75 F0                 RP              ;NOPE!
FE76 3EFF               MVI     A,GBLNK ;=GRAPHICS BLANK
FE78 12                 STAX    D
FE79 C9                 RET
                                ;CBLNK CONVERTS SCREEN ADDRESSES & BLANKS THE CELL:
FE7A CD70FE     CBLNK:  CALL    CNVTX
FE7D E6FF               ANI     GBLNK   ;=BLANK
FE7F F680               ORI     80H
FE81 B1                 ORA     C
FE82 C9                 RET
                ;CNVRT  CHANGES <HL> COORDINATE INTO ADDRESS
                ;       AND     GENERATES BIT MASK. <A>=MEMORY CONTENTS
FE83 C5         CNVRT:  PUSH    B
                        ;NORMALIZE THE COORDINATES
FE84 7D                 MOV     A,L     ;FIRST THE X-COORDINATE
FE85 E67F               ANI     7FH
FE87 6F                 MOV     L,A
                                        ;AND NOW THE Y-COORDINATE
FE88 3A0B00             LDA     VDMCW   ;HOW MANY
FE8B E603               ANI     3       ;SCREEN LINES
FE8D 47                 MOV     B,A     ;ARE PROTECTED?
FE8E 3E30               MVI     A,48    ;CONVERT
FE90 90                 SUB     B       ;LINES
```

```
FE91 90                 SUB     B       ;TO
FE92 90                 SUB     B       ;GRAPHICS
FE93 47                 MOV     B,A     ;CELLS
FE94 7C                 MOV     A,H     ;AND NORMALIZE
FE95 90         D1:     SUB     B       ;MODULO <B>
FE96 F295FE             JP      D1      ;= 48-3*LINES
FE99 80         D2:     ADD     B       ;ALL
FE9A FA99FE             JM      D2      ; DONE!
FE9D 67                 MOV     H,A     ;  SO SAVE IT
FE9E E5                 PUSH    H
FE9F 44                 MOV     B,H     ;THE
FEA0 4D                 MOV     C,L     ; FOLLOWING
FEA1 5C                 MOV     E,H     ;  COMPLEX
FEA2 1600               MVI     D,0     ;   LOGIC
FEA4 210100             LXI     H,1     ;    CONVERTS
FEA7 19                 DAD     D       ;FROM
FEA8 29                 DAD     H       ; SCREEN
FEA9 29                 DAD     H       ;  ADDRESS
FEAA 19                 DAD     D       ;   IN <HL>
FEAB 29                 DAD     H       ;TO
FEAC 29                 DAD     H       ; MEMORY
FEAD 19                 DAD     D       ;  ADDRESS
FEAE 54                 MOV     D,H     ;   IN <DE>
FEAF 7D                 MOV     A,L
FEB0 E6C0               ANI     0C0H
FEB2 5F                 MOV     E,A
FEB3 19                 DAD     D
FEB4 19                 DAD     D
FEB5 29                 DAD     H
FEB6 29                 DAD     H
FEB7 78                 MOV     A,B
FEB8 94                 SUB     H
FEB9 47                 MOV     B,A
FEBA 3EC0               MVI     A,0C0H  ;LOW BYTE OF BOTTOM LINE
FEBC 93                 SUB     E
FEBD 5F                 MOV     E,A
FEBE 3ECF               MVI     A,VDEND-1 ;HI BYTE OF BOTTOM LINE
FEC0 9A                 SBB     D
FEC1 57                 MOV     D,A
FEC2 79                 MOV     A,C
FEC3 1F                 RAR
FEC4 B3                 ORA     E
FEC5 5F                 MOV     E,A
FEC6 79         GENR8:  MOV     A,C     ;GENERATE CELL'S BIT MASK
FEC7 1F                 RAR
FEC8 78                 MOV     A,B
FEC9 17                 RAL
FECA 4F                 MOV     C,A
FECB 0600               MVI     B,0
FECD 21D7FE             LXI     H,DTAB
FED0 09                 DAD     B
FED1 7E                 MOV     A,M
FED2 E1                 POP     H
FED3 C1                 POP     B
FED4 4F                 MOV     C,A
FED5 1A                 LDAX    D
FED6 C9                 RET
FED7 04         DTAB:   DB      04H     ;LOWER LEFT    CELL
FED8 20                 DB      20H     ;LOWER RIGHT     BIT
FED9 02                 DB      02H     ;MIDDLE LEFT      MASKS
```

```
FEDA 10                 DB      10H     ;MIDDLE RIGHT       IN 2x3
FEDB 01                 DB      01H     ;TOP LEFT            CHARACTER
FEDC 08                 DB      08H     ;TOP RIGHT            SPACE
                ;-------------------------------------------------------------;
                ;       GRAPHICS        ROUTINES CALLABLE FROM BASIC:         ;
                ;  ENTER WITH DATA IN <DE>, RETURN FROM <HL> AS PER           ;
                ;  THE CONVENTIONS IN MOST BASIC MACHINE LANGUAGE CALLS.      ;
                ;  THE CALLS SHOULD LOOK SOMETHING LIKE:                      ;
                ;           R = CALL ( ENTRY POINT , 256*Y+X )                ;
                ;-------------------------------------------------------------;
                ;       CHECK   ROUTINE. RETURNS ZERO IF CELL IS BLACK,
                                        ; 255 IF WHITE, OR ASCII
FEDD EB         BASC:   XCHG    ;EQUIV. OF CHARACTER IF CELL OCCUPIED BY TEXT
FEDE CD83FE             CALL    CNVRT
FEE1 210000             LXI     H,0
FEE4 FE80               CPI     80H     ;IS <A> GRAPHICS OR TEXT?
FEE6 F2EBFE             JP      BASC2
FEE9 6F         BASC1:  MOV     L,A
FEEA C9                 RET             ;WITH   ASCII CHARACTER
FEEB A1         BASC2:  ANA     C
FEEC C0                 RNZ             ;IF BLACK, RETURN ZERO
FEED 2D                 DCR     L       ;IF WHITE RETURN 255
FEEE C9                 RET
FEEF EB         BASW:   XCHG    ;PLOT   WHITE DOT, RETURN <HL>=<YX> SCREEN COORDINATES
FEF0 CD65FE             CALL    WHITE
FEF3 C9                 RET
FEF4 EB         BASB:   XCHG    ;PLOT   BLACK DOT, RETURN <HL>=<YX> SCREEN COORDINATES
FEF5 CD6BFE             CALL    BLACK
FEF8 C9                 RET
FEF9 EB         BASA:   XCHG    ;RETURN <HL>=SCREEN MEMORY ADDRESS FOR THE CELL
FEFA CD83FE             CALL    CNVRT
FEFD EB                 XCHG
FEFE C9                 RET
```

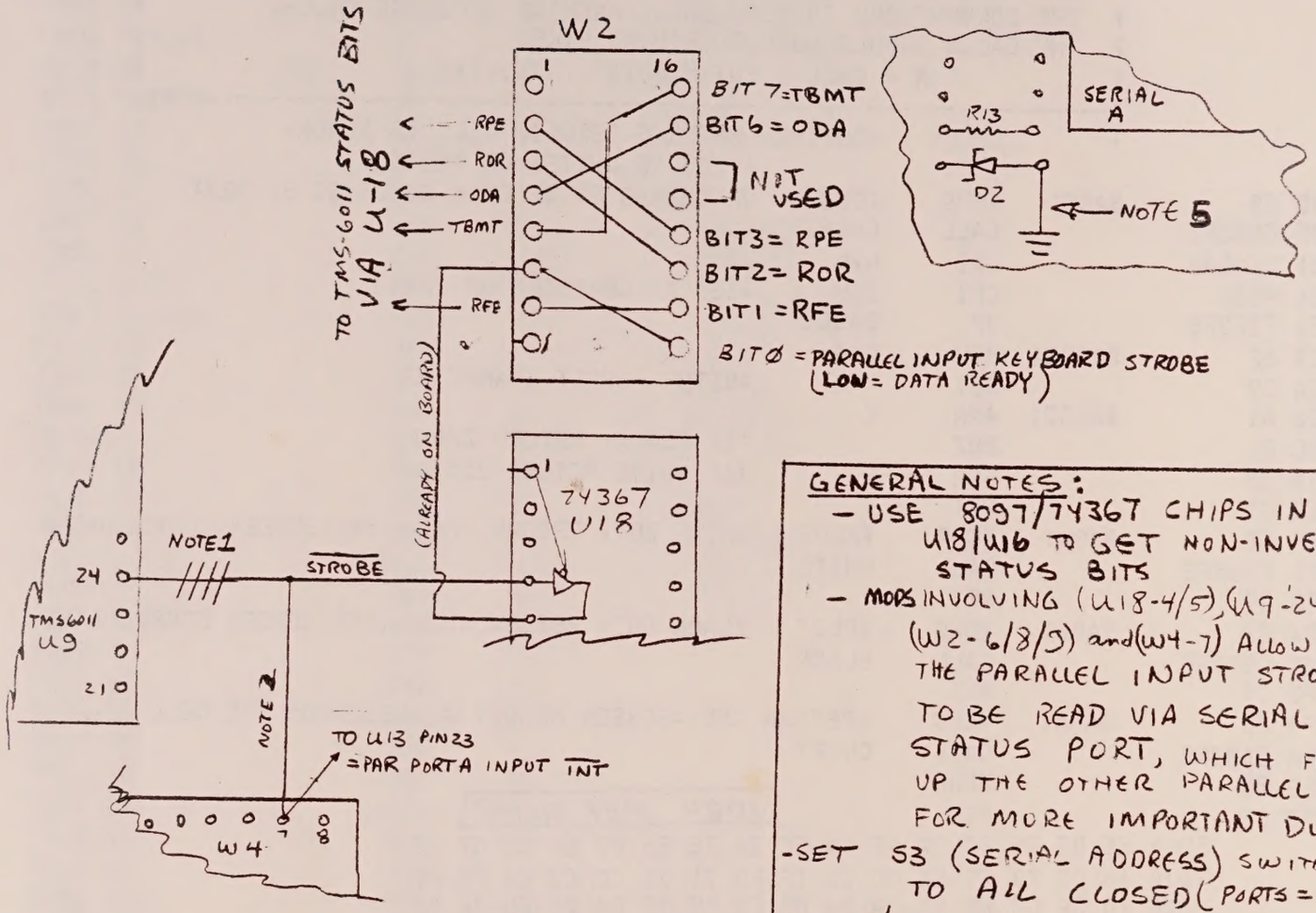
VDB4 HEX DUMP

```
FD00 E5 D5 C5 F5 2A 0E 00 5E 23 7E E6 03 C6 CC 57 23
FD10 46 EB 70 79 FE 0C C2 1F FD 21 00 CC C3 C1 FD FE
FD20 0D CA CC FD FE 0A CA D3 FD FE 0B CA BE FD 3A 0B
FD30 00 E6 04 79 C2 89 FD FE 15 C2 42 FD 01 C0 FF C3
FD40 A9 FD FE 04 C2 4D FD 01 40 00 C3 A9 FD FE 08 CA
FD50 57 FD FE 5F C2 5D FD 01 FF FF C3 A9 FD FE 06 CA
FD60 A6 FD FE 01 C2 6D FD CD 55 FE C3 AA FD FE 09 C2
FD70 7E FD 36 20 23 7D E6 07 C2 72 FD C3 AA FD FE 07
FD80 C2 89 FD CD F0 FC C3 AA FD 7D E6 3F FE 3F C2 A5
FD90 FD 3A 0B 00 E6 40 C2 AA FD D5 23 7C FE D0 CA DE
FDA0 FD CD 35 FE D1 71 01 01 00 09 7C E6 03 C6 CC 67
FDB0 7E 36 AD EB 77 2B 72 2B 73 F1 C1 D1 E1 C9 CD 55
FDC0 FE E5 36 20 23 7C FE D0 FA C2 FD E1 7D E6 C0 6F
FDD0 C3 AA FD D5 11 40 00 19 7C FE D0 C2 45 FE 3A 0B
FDE0 00 17 DA 0D FE 21 FF CF 36 80 CD F1 FD D1 C3 BE
FDF0 FD CD 03 F0 FE 20 C8 FE 2F CA 09 FE FE 2D C2 4C
FE00 FE 3A 0B 00 F6 80 32 0B 00 D1 36 20 23 E5 CD 55
FE10 FE EB 21 40 00 19 7E 12 13 23 7C FE D0 C2 16 FE
FE20 21 C0 CF CD 35 FE E1 11 C0 FF 19 D1 79 FE 20 FA
FE30 AA FD C3 A5 FD E5 7D E6 C0 6F 36 20 23 7D E6 3F
FE40 C2 3A FE E1 C9 CD 35 FE D1 C3 AA FD 3A 0B 00 F6
FE50 20 32 0B 00 C9 26 CC 3A 0B 00 0F 0F E6 C0 6F C9
FE60 CD 70 FE A1 C9 CD 7A FE A9 12 C9 CD 7A FE 12 C9
FE70 CD 83 FE FE 80 F0 3E FF 12 C9 CD 70 FE E6 FF F6
FE80 80 B1 C9 C5 7D E6 7F 6F 3A 0B 00 E6 03 47 3E 30
FE90 90 90 90 47 7C 90 F2 95 FE 80 FA 99 FE 67 E5 44
FEA0 4D 5C 16 00 21 01 00 19 29 29 19 29 29 19 54 7D
FEB0 E6 C0 5F 19 19 29 29 78 94 47 3E C0 93 5F 3E CF
FEC0 9A 57 79 1F B3 5F 79 1F 78 17 4F 06 00 21 D7 FE
FED0 09 7E E1 C1 4F 1A C9 04 20 02 10 01 08 EB CD 83
FEE0 FE 21 00 00 FE 80 F2 EB FE 6F C9 A1 C0 2D C9 EB
FEF0 CD 65 FE C9 EB CD 6B FE C9 EB CD 83 FE EB C9 FF
```

CD xx xx = BELL ROUTINE AT xx xx

OR

00 00 00 = NO BELL ROUTINE INCLUDED



GENERAL NOTES:

- USE 8097/74367 CHIPS IN U18/U16 TO GET NON-INVERTED STATUS BITS
- MODS INVOLVING (U18-4/5), (U9-24), (W2-6/8/5) and (W4-7) ALLOW THE PARALLEL INPUT STROBE TO BE READ VIA SERIAL STATUS PORT, WHICH FREES UP THE OTHER PARALLEL PORT FOR MORE IMPORTANT DUTIES.
- SET S3 (SERIAL ADDRESS) SWITCH TO ALL CLOSED (PORTS = Ø-3)
- SET S1/S2 "PR" OPEN (STATUS = PORTS Ø & 2, DATA = PORTS 1 & 3). OTHER S1/S2 SWITCHES ARE TERMINAL DEPENDENT.

SOLID-STATE MUSIC IO-4 BOARD MOD'S AND STRAPPING FOR EASE OF USE VIA AMS-80 WITH PARALLEL KEYBOARD

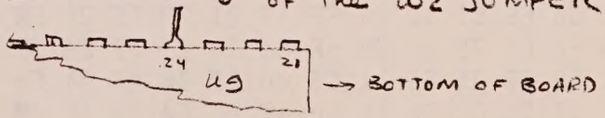
T. CLARK

Revised 23 MAY 78
28 AUG 79

BOARD MODIFICATION NOTES:

(1) DISABLE TEOC ON TMS6011/AY5-1013 UART (U9) BY BENDING OR CUTTING PIN 24 SO THAT NO CONTACT IS MADE WITH THE SOCKET. THIS FREES UP ONE STATUS BIT ON PIN 6 OF THE W2 JUMPER

e.g.



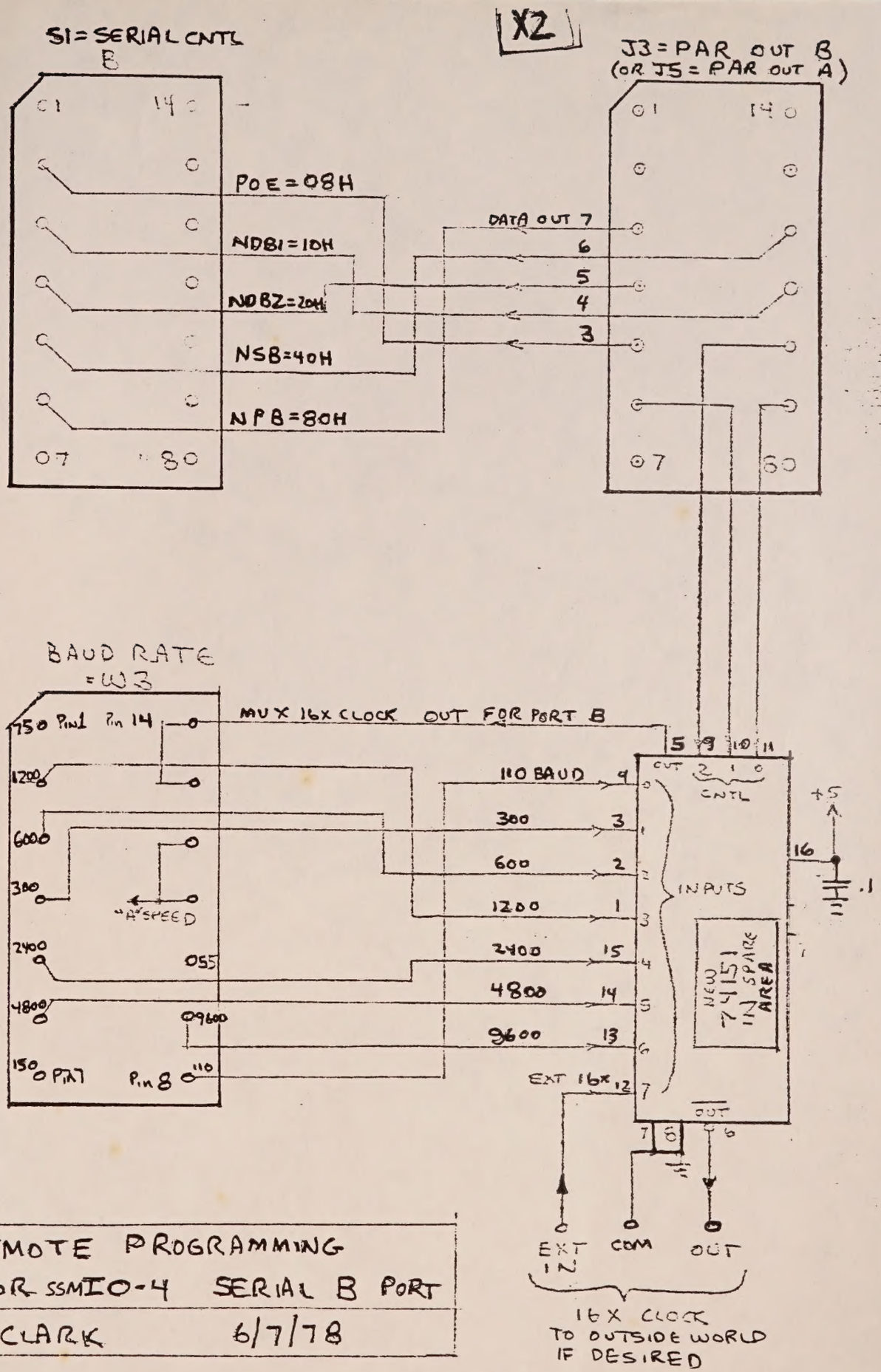
(2) ADD JUMPER FROM EITHER (U9 PIN 24) OR (U18 PIN 4) TO (W4 PIN 7)

(3) SOFTWARE WHICH USES THIS MODIFICATION FOR A PARALLEL INPUT KEYBOARD IS GIVEN ON PAGE C-11, LINES 3310-3500. THE SERIAL I/O SOFTWARE ON PAGES C9-C11 USES THE W1/W2 STRAPPING SHOWN ABOVE.

(4) "B" PORTS MAY BE MODIFIED THE SAME WAY, WITH W1→W2, U8→U9, U16→U18, AND (W4 PIN 6)→(W4 PIN 7)

(5) SOME IO4 BOARDS HAVE TRACE MISSING FROM ANODE OF D2 to GND. ADD IT!

X2



REMOTE PROGRAMMING FOR SSMIO-4 SERIAL B PORT	
T. CLARK	6/7/78